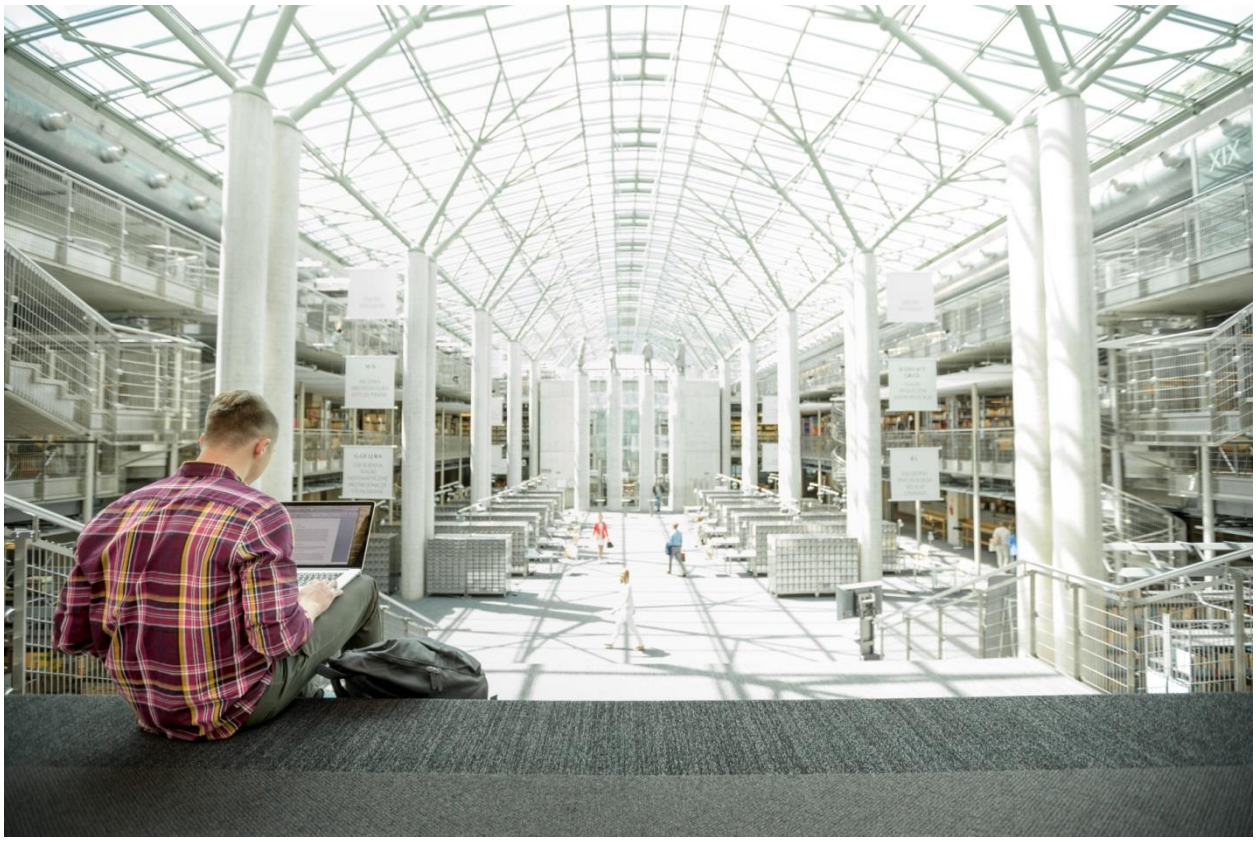


FlexPod Datacenter with IBM Cloud Private

FlexPod Datacenter with IBM Cloud Private, Cisco UCS Manager, NetApp ONTAP 9.3 AFF, and VMware vSphere 6.5U1

Last Updated: August 14, 2018

CISCO VALIDATED DESIGN

About the Cisco Validated Design Program

The Cisco Validated Design (CVD) program consists of systems and solutions designed, tested, and documented to facilitate faster, more reliable, and more predictable customer deployments. For more information, visit:

http://www.cisco.com/go/designzone.

Table of Contents

Executive Summary

Error! Reference source not found.Error! Reference source not found.Error! Reference source not found.Error! Reference source not found.IBM Cloud Private (ICP) is an on-premises platform for developing and managing containerized applications for cloud-native and application-modernization use cases. It is an integrated environment built on Kubernetes as its container orchestration, and includes a private image repository for Docker containers, a management console, a monitoring framework, many open source based, as well as IBM containerized applications, and more. Combining ICP with FlexPod, the converged infrastructure from Cisco and NetApp, can simplify the deployment and the management of your infrastructure. You can also benefit from improved storage efficiency, better data protection, lower risk, and the flexibility to scale this highly available enterprise-grade infrastructure stack to accommodate new business requirements and other changes over time.

This combined solution helps organization accelerate their digital transformation journey. For many companies, this means adopting micro-services and DevOps practices and moving away from monolithic application development practices as a first step in creating more business value from their applications.

To help customers, business partners, and other interested parties with their digital transformation and to enhance their cloud-native and application modernization practices, Cisco, NetApp, and IBM partnered to produce this Cisco Validated Design (CVD) for the FlexPod for IBM Cloud Private solution. This document provides a reference architecture that includes design and deployment guidance, best practices, and other recommendations. With this guide, one can simplify the digital transformation and address many of the associated challenges, including cost and risk reduction, scalability, operational flexibility, and security.

Business Challenges

There are many challenges along the journey of the digital transformation that organizations have to deal with, and the task is even more difficult as external pressures from competitors and customers require businesses to act fast, while IT organizations, that traditionally carry most of the burden, are usually only staffed to keep the lights on. To succeed, organizations must adopt new technologies, such as the cloud, by updating existing IT infrastructure to accommodate new requirements, including those related to security and data privacy. They must implement new processes and methodologies such as DevOps and micro-services, and they need to address gaps in their current available skill sets.

The need to move fast along the digital transformation path leaves organizations with very little room for errors when making decisions about their infrastructure and its related services. The speed of transformation further increases the exposure to risks associated with any changes.

Even though cloud-based services can accelerate the digital transformation that customers are seeking, using public cloud-based services exclusively can be difficult, and a complete transition might take a significant amount of time. In some cases, it is simply not possible because of regulations or other constraints related to data sensitivity, privacy, and security.

New models of operation and new ways to consume services often come with unknown cost factors that all organizations want to avoid because a successful business plan typically depends on predictable and controlled costs.

Solution

The FlexPod for ICP solution addresses many of these challenges because it allows organizations to move faster with less risk. This solution provides more flexibility and options for managing, operating, and scaling IT infrastructures along the digital transformation journey, while operating in a predictable and economical cost structure.

The FlexPod for ICP solution covered in this Cisco Validated Design (CVD) is a complete converged infrastructure (CI) stack with a software platform from IBM. The converged infrastructure is comprised of Cisco compute and network switches, a NetApp ONTAP Storage and Data Management platform, and VMware vSphere as the underlying hypervisor. This stack is integrated so customers do not have to allocate time and resources researching the required components and how to optimally integrate them to support ICP. This CI solution was also tested and validated for ICP, which saves time and minimizes risks so that you can quickly deploy the solution, become productive in days, and focus on your business rather than worrying about setting up infrastructure.

The ICP software platform offers a sound value proposition to customers because IBM has included all the necessary tools to manage and provision a private cloud. Therefore, customers can start cloud-native or application modernization development very quickly without needing to research the setup and integration of the various components in the software stack. Although the platform is based on open source container technologies (Docker and Kubernetes), all necessary components are already packaged and integrated together. This includes crucial tools for monitoring, logging, alerting, and metering. It also includes ready-out-of-the-box containers such as Jenkins for Continuous Integration and Continuous Delivery (CI/CD), WebSphere Liberty, MongoDB, and other containerized applications. This integration allows customers to focus on development efforts and the consumption of services in an on-premises cloud model, hence adhering to any data related regulations or sensitivities that might limit consumption of public cloud based services.

Another key aspect of the solution is minimizing risks by providing more flexibility. FlexPod is a highly flexible and scalable CI platform that is based on a unified architecture. With FlexPod, you can start small and grow as your business needs grow. FlexPod also support various connectivity and integration methods in your existing infrastructure that enable you to replicate data to other sites, including the cloud.

Solution Overview

Introduction

The featured FlexPod Datacenter for ICP solution is a designed, integrated, and validated architecture for a data center that combines Cisco UCS servers, the Cisco Nexus family of switches, and NetApp storage arrays into a single, flexible architecture. FlexPod is designed for high availability (HA), with no single point of failure, while maintaining cost-effectiveness and flexibility in the design to support a wide variety of workloads. The FlexPod solution covered in this document is for IBM Cloud Private (ICP), an on-premises development platform built on Kubernetes.

This is classified as a FlexPod Datacenter solution and, in addition to the hardware components, includes VMware vSphere as the hypervisor layer on top of which the ICP components are deployed.

Integration between ICP and the storage and data management services occurs at several levels, all of which are captured in the design aspects of this document. The main storage integration is based on a Kubernetes plug-in called Trident, which is a dynamic storage provisioner from NetApp. After Trident is installed, it becomes part of the Kubernetes framework, which helps accelerates development and complements DevOps practices.

The ICP software is installed on top of a Kubernetes cluster with the nodes running as Linux Ubuntu VMs on ESXi hosts deployed on the Cisco UCS platform.

The following design and deployment aspects for the FlexPod for ICP solution are covered in this document:

- IBM Cloud Private 2.1.02 Enterprise Edition
- FlexPod Datacenter converged infrastructure
- VMware vSphere 6.5 u1
- NetApp Trident – Dynamic storage provisioner for Kubernetes

The document also covers key configurations based on the tested environment and best practices.

Audience

The intended audience for this document includes, but is not limited to, DevOps managers, IT infrastructure managers, application development leaders, business digital transformation leaders, storage and data management managers, sales engineer and architects working with hybrid and private clouds, and other parties that are looking for a tested, market-proven CI solution that offers flexibility and simplicity in support of their cloud native and application modernization needs along their digital transformation journey.

Purpose of This Document

The purpose of this design and deployment guide is to provide a reference architecture with specific examples indicating how the solution was designed, deployed, and tested. In addition, the document provides several best practices and recommendations that simplify your implementation of this solution.

What’s New?

Although the FlexPod for ICP solution was developed based on an existing CVD (FlexPod Datacenter with VMware vSphere 6.5 Design Guide - July 2017), the following elements distinguish this version of FlexPod from previously published solutions:

- This solution design is optimized for IBM Cloud Private. It addresses HA, scalability, and operational best practices that reduce associated risks and accelerate cloud-native and application modernization use cases.

- Integration with Kubernetes for dynamic persistent storage provisioning simplifies operations within the framework used by the end users.
- Advanced data management accelerates development cycles and complements microservices and DevOps practices.
- Multiple deployment models showcase the flexibility and scalability of the FlexPod solution leading to a better ROI.

For more information about previous FlexPod designs, see: http://www.cisco.com/c/en/us/solutions/design-zone/data-center-design-guides/flexpod-design-guides.html

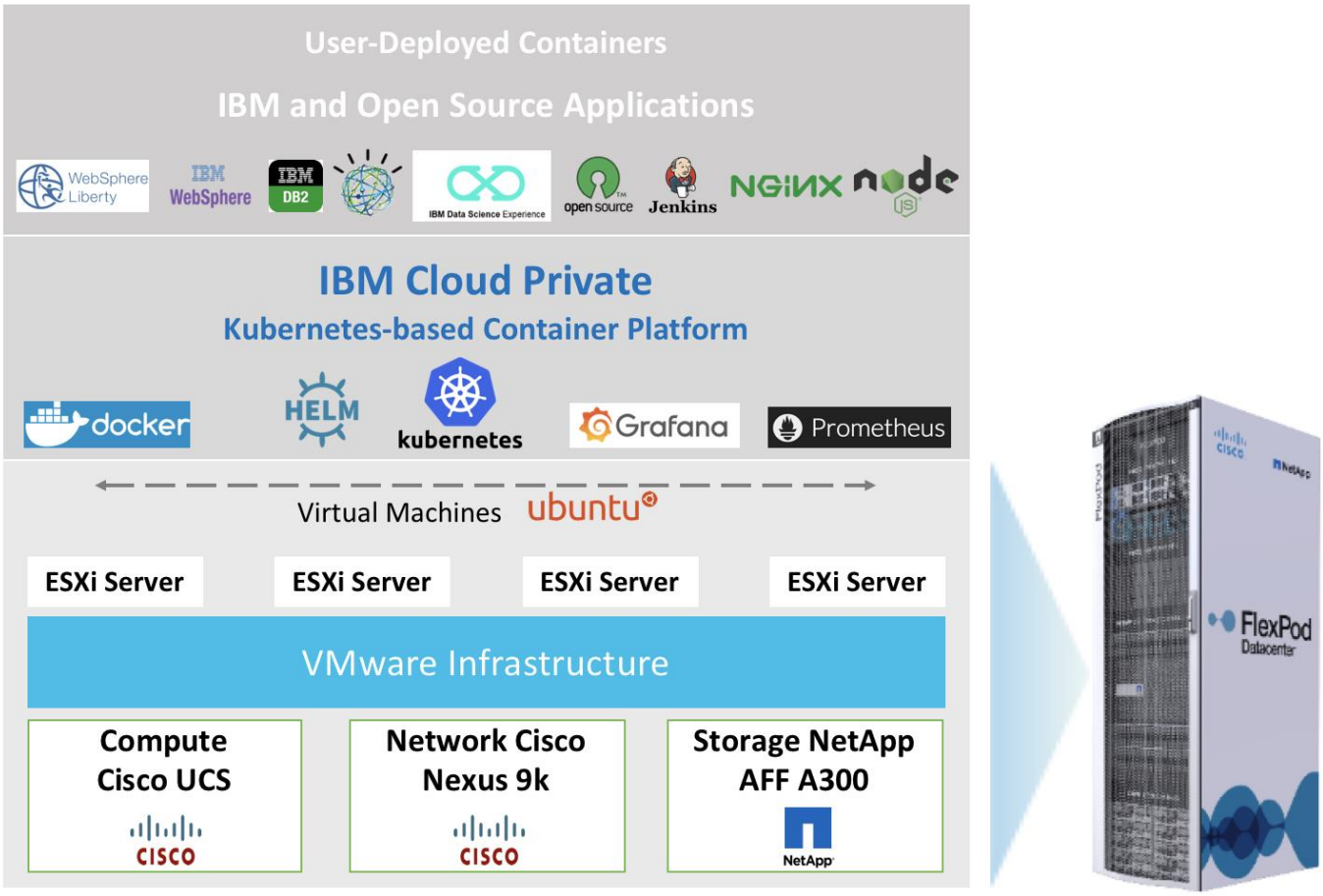
Solution Summary

Successful digital transformation requires a complete solution that enables DevOps and data science to work closely with infrastructure administrators to maintain data sovereignty and simplify enterprise management. The solution should also help modernize existing applications, develop new cloud-native solutions, and leverage cloud-based services. FlexPod combined with the ICP platform provides faster deployment, higher efficiency, a highly scalable and flexible stack, and less risk. These intrinsic capabilities improve developer productivity, reduce build times, and optimize storage with advanced data management capabilities.

This solution includes a hardware stack from Cisco and NetApp, a hypervisor from VMware, and ICP software platform from IBM, a set of tools for integration and management, and a collection of containerized applications available out-of-the-box (Figure 1). These components are integrated so that customers can deploy the solution quickly and economically while eliminating many of the risks associated with researching, designing, building, and deploying similar solutions from the ground up.

This document addresses the cloud-native and application-modernization use cases for ICP. It also provides references on how to use FlexPod for additional services beyond ICP to maximize operational value and also maximize other aspects of ROI when integrating the platform into existing infrastructure.

Figure 1 FlexPod for IBM Cloud Private



Solution Benefits

This joint solution offers the following benefits:

1. Accelerated time to market and productivity
2. Operational cost reduction
3. Business risk reduction and flexibility to accommodate future requirements
4. Enterprise-grade infrastructure features
5. Advanced data management
6. Infrastructure scalability and flexibility

IBM Cloud Private helps customers accelerate their digital transformation because it contains all the required tools needed out-of-the-box for consuming on-premises, cloud-based services. Customers can develop and deploy cloud native applications and modernize existing IBM middleware Java EE-type heritage applications. You do not need to spend resources researching how to integrate components into a cohesive platform that you can easily utilize, manage, and support.

In addition, IBM has developed a common CLI that provides a unified user experience for its public cloud services in conjunction with ICP as an on-premises platform. If needed, the platform can support services from other cloud

providers by leveraging the included Cloud Automation Manager (CAM). Beyond the well-integrated software stack at the ICP platform level, FlexPod offer the same value at the hardware stack level.

The following are additional specific benefits of the joint solution:

- Highly available, scalable platform
- Flexible architecture supports various deployment models
- Accelerated adoption of new technology to modernize heritage applications and cloud native development
- A trusted platform from industry leaders, Cisco and NetApp
- Converged infrastructure validated for ICP with a published CVD that includes best practices and recommendations, which minimizes risks and accelerates development cycles
- Ecosystem of partners and the recognized and trusted FlexPod brand
- Cooperative support between NetApp, Cisco, IBM, VMware, and Canonical
- Easy to deploy, consume, and manage; saves time and resources on research, procurement, and integration
- Enterprise-grade, highly available infrastructure stack
- Flexible and highly scalable architecture with outstanding application performance
- Superior data management and storage efficiencies
- Advanced data protection capabilities
- Integration with VMware
- Integration with Kubernetes for dynamic provisioning of persistent storage services with advanced data management capabilities
- Enhanced CI/CD workflow and practices associated with DevOps and micro-services
- Comprehensive set of APIs (including REST) for automation and integration with existing orchestration and management tools

Solution Components

The solution offers redundant architecture from a compute, network, and storage perspective. The solution consists of the following key components:

- NetApp AFF A300 storage controllers
  - High Availability (HA) pair in switchless cluster configuration
  - 40GbE adapter used in the expansion slot of each storage controller
  - ONTAP 9.3 storage OS
- Cisco Nexus 9332PQ switches
  - Pair of switches in vPC configuration
- Cisco UCS 6332-16UP Fabric Interconnect
  - 40Gb Unified Fabric
- Cisco UCS 5108 Chassis
  - Cisco UCS 2304 IOM

  - Cisco UCS B200 M5 servers with VIC 1340
  - Cisco UCS C220 M4 servers with VIC 1385
- IBM Cloud Private 2.1.02 Enterprise
- NetApp Trident 18.04
- VMware vSphere 6.5 u1

Use Cases

Cloud-Native, Application Modernization and a model of hosting additional services such as enterprise applications are covered as use cases in this document. These use cases reflect the flexibility of the FlexPod platform to accommodate various requirements and enable the different workloads and topologies you might focus on.

Cloud Native

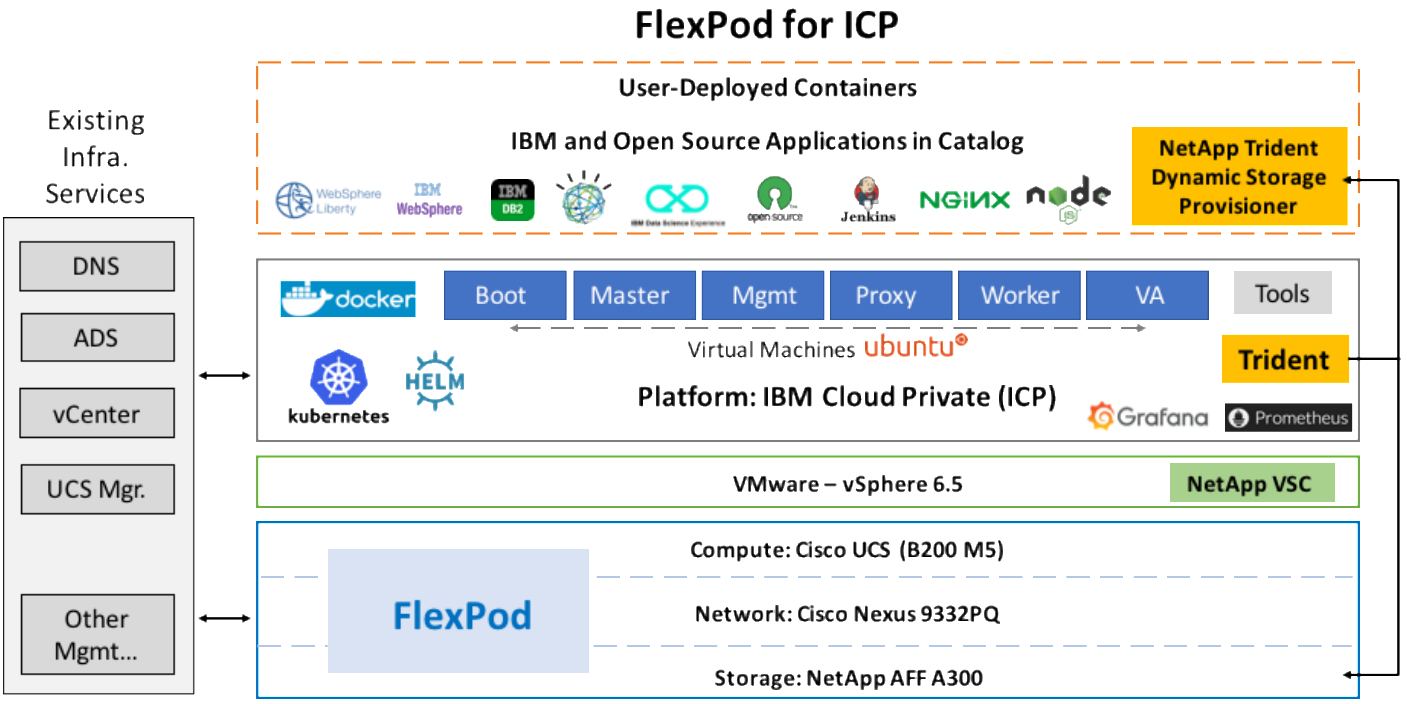
Being an on-premises solution, FlexPod for ICP is commonly deployed in environments with existing infrastructure components and services already available and running. Under these circumstances, the solution must be integrated with existing components such as DNS, directory services, VMware vCenter and other management and orchestration tools.

In some cases, services that are required for running ICP are not available within the existing on-premises environment, and you must install these services as prerequisites for the ICP platform. Installing and configuring these services are not within the scope of this CVD.

ICP is cloud-native ready out-of-the-box. No additional software packages or tools are required, but you can add to the platform-containerized applications with your own tools leveraging the image repository option for easy access.

Figure 2 provides a high-level overview of the FlexPod for ICP platform with the cloud-native use case.

Figure 2 FlexPod for ICM – Cloud Native Topology



Application Modernization and Enterprise Applications

FlexPod for ICP supports enterprise applications in several ways. First, it is easy to integrate the FlexPod platform with the existing on-premises infrastructure for consistent management of the entire end-to-end infrastructure. With many customers already having enterprise and middleware applications deployed, prior to introducing ICP to their environment, having the FlexPod integrated with the existing infrastructure will further simplify the implementation of ICP in the combined environment also with the existing enterprise and middleware applications.

The other common approach is to host any enterprise and middleware applications that were newly purchased with ICP (ICP Enterprise Edition with the Enterprise bundle) on the FlexPod platform itself.

The software of the enterprise bundle are not included with the ICP software package; they must be installed separately after the installation of ICP and can be added to the ICP catalog (Helm charts). The enterprise bundle includes the following components:

- WebSphere Application Server Network Deployment (WAS ND)

  IBM WebSphere Application Server Network Deployment provides an advanced, flexible runtime environment for large-scale application deployments. It offers near-continuous availability with advanced performance and management capabilities for mission-critical applications.

- Db2 Direct Advanced

  IBM DB2 Direct Advanced is the transformation data platform for transactional and analytical workloads in the era of digital, cloud, and cognitive. It provides continuous availability of data to keep transactional and analytics operations at maximum efficiency to ensure access to data is not impacted by any planned or unplanned downtime

- MQ Advanced

  IBM MQ Advanced is a robust messaging middleware solution that provides simple, secure, scalable and reliable messaging to connect and integrate applications, systems and services within and between multiple platforms, including both on premise environments and cloud deployments.

- Microclimate (formerly Micro-services Builder)

  Microclimate provides end-to-end user experience for developing and deploying modern apps and accessible by default in IBM Cloud Private. Microclimate provides a first class experience for deployment and management of containerized apps.

- API Connect Professional

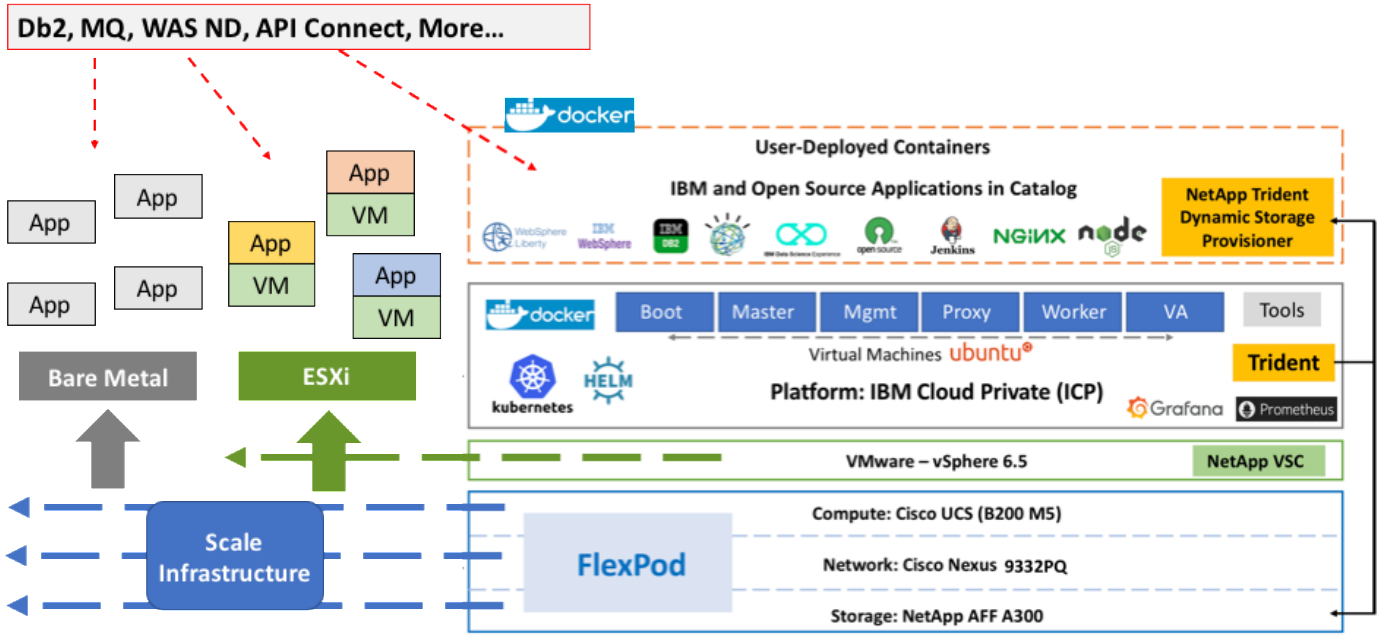
  IBM API Connect Professional provides a comprehensive solution to manage your entire API lifecycle from creation to management.

- Urban Code Deploy

  IBM UrbanCode Deploy is an application release automation solution that combines robust visibility, traceability and auditing capabilities.

Customers commonly add new applications and tools to the FlexPod ICP platform and also want to integrate with products that are already in place. For example, the organization might use a non-IBM database (say, Oracle) and might prefer to keep this database in place and allow the containerized applications running within the ICP platform to interact with the database. They might also use other IBM middleware products. IBM offers a Docker container version of many of its enterprise applications, including MQ, WebSphere, and several products in the Db2 family. However, some organizations might not deploy these products in their container version and might start with a more traditional approach of hosting the application on a guest OS running as a VM, and in some cases the application may end up being deployed directly on the server (bare metal); FlexPod can be extended to host these applications as well (Figure 3).

Figure 3 Hosting Enterprise Applications on FlexPod for ICP



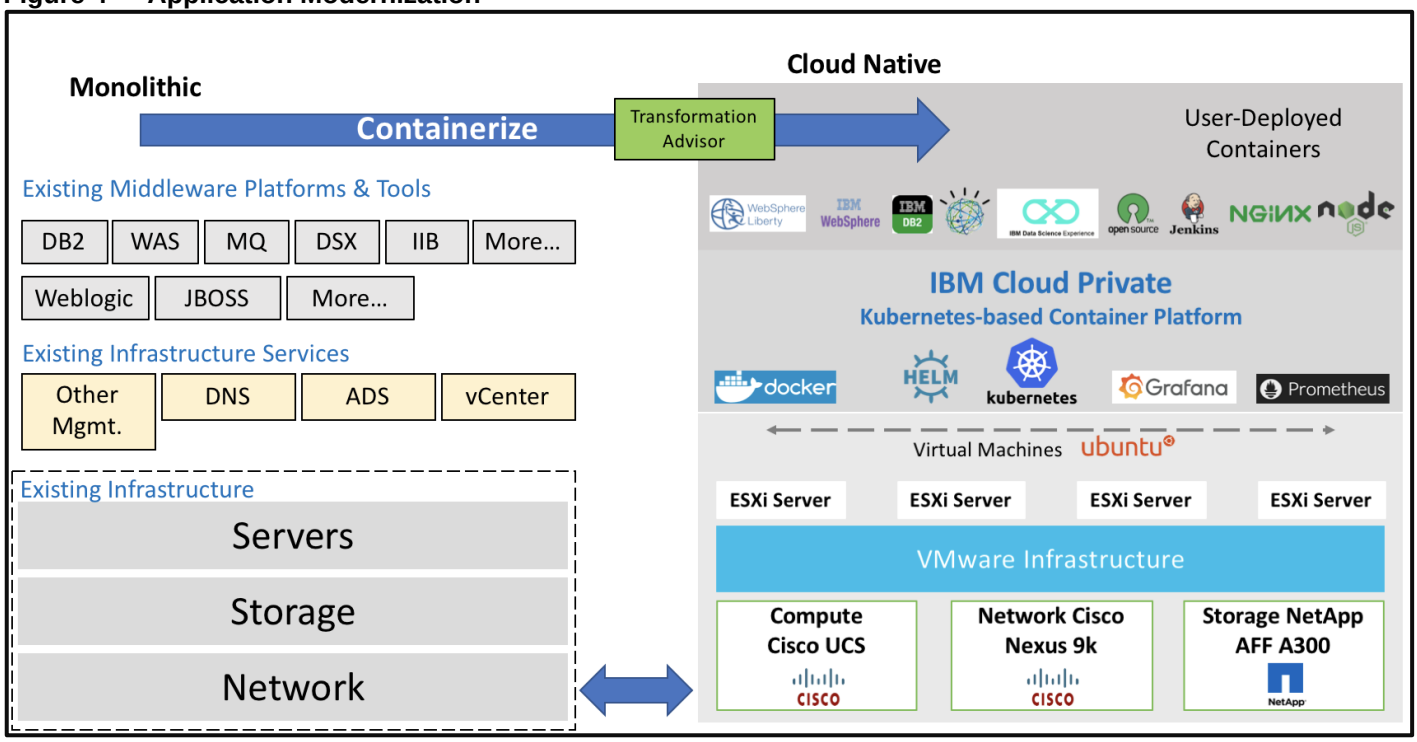
Additional workloads and applications beyond ICP can be hosted on the FlexPod CI platform. This deployment model can also accommodate IBM enterprise applications that can run on a VM as part of the vSphere environment or in a bare metal fashion directly on the UCS servers if they are not containerized. Storage capacity or additional controllers can also be added non-disruptively, in a manner similar to servers and network port capacity, to support additional data requirements. The flexibility of the platform to scale improves ROI and other aspects of productivity and, in addition, offers you more confidence that the platform supports your changing needs over time.

Regarding application modernization, each customer develops their own plan for transforming monolithic applications to a cloud-native topology. The plan reflects an individual company's business needs, including pressures from customers, competitors, regulations, and other issues. A plan can also address gaps in current skill

sets, the complexity of applications, data privacy and other data-related issues, existing tools, infrastructure readiness, operational and other cost factors, and so on. The process can be very complex. Therefore, a reliable, flexible, and scalable platform at the infrastructure level that complements the value proposition of the ICP software platform can have tremendous value.

As explained earlier, FlexPod can host containerized application that run on the ICP platform, and it can also host applications running as a VM or as bare metal on the Cisco UCS servers. IBM provides tools to help with the modernization process of applications, including Transformation Advisor (TA). To help convert an application to a cloud-native model, Transformation Advisor scans the code of existing Java applications, collects data, and analyzes the readiness of an application for conversion to a Docker container. Transformation Advisor can be deployed as a container in ICP.

Figure 4 Application Modernization



Technology Overview

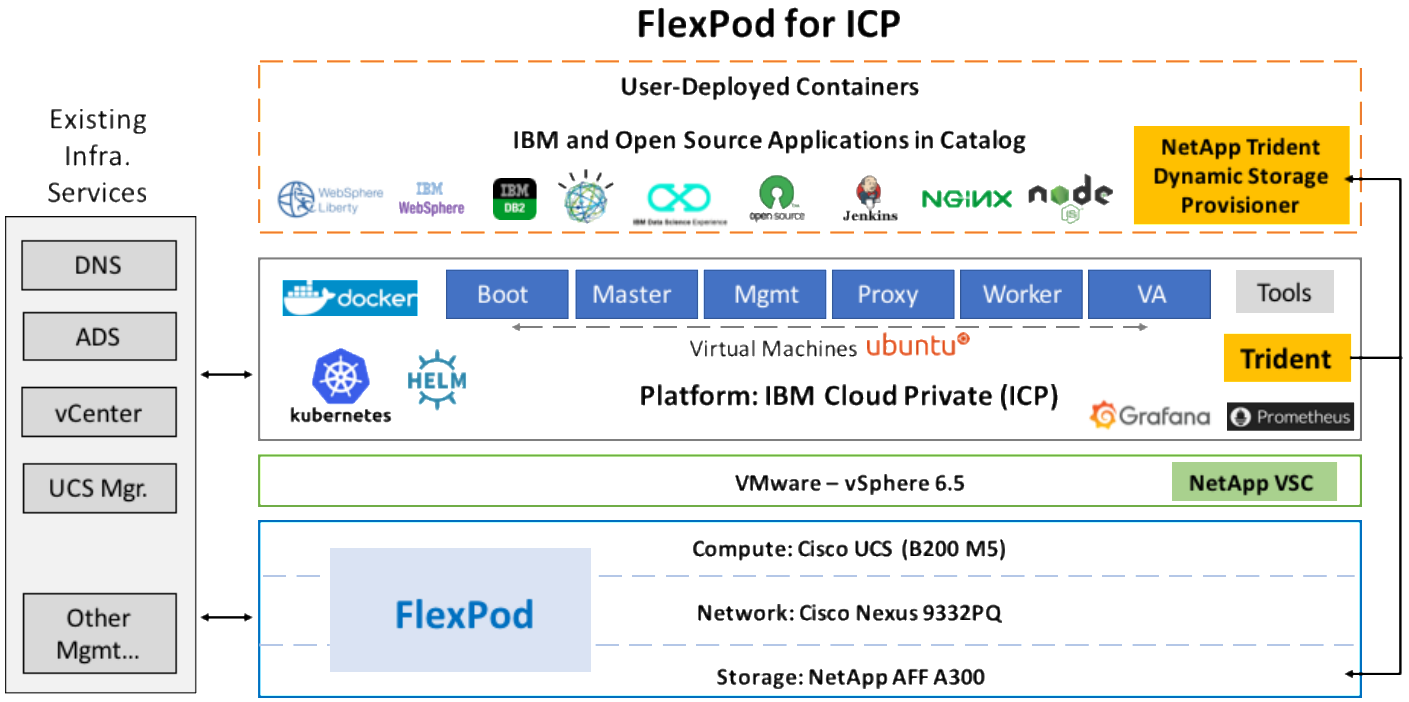
FlexPod Datacenter for IBM Cloud Private consists of following components:

- FlexPod converged infrastructure (software and system hardware products from NetApp and Cisco)
- IBM Cloud Private (software)
- NetApp Trident, a dynamic Storage Provisioner for Kubernetes (software)

Figure 5 illustrates the high-level architecture of the solution.

As indicates in the diagram, the ICP platform itself is deployed on a Kubernetes cluster with a set of virtual machines that in addition to carrying the various roles of the platform's services, they also host myriad of tools for managing the platform itself and for managing activities associated with containerized applications deployed by the end users.

Figure 5 High-Level Architecture



FlexPod Converged Infrastructure

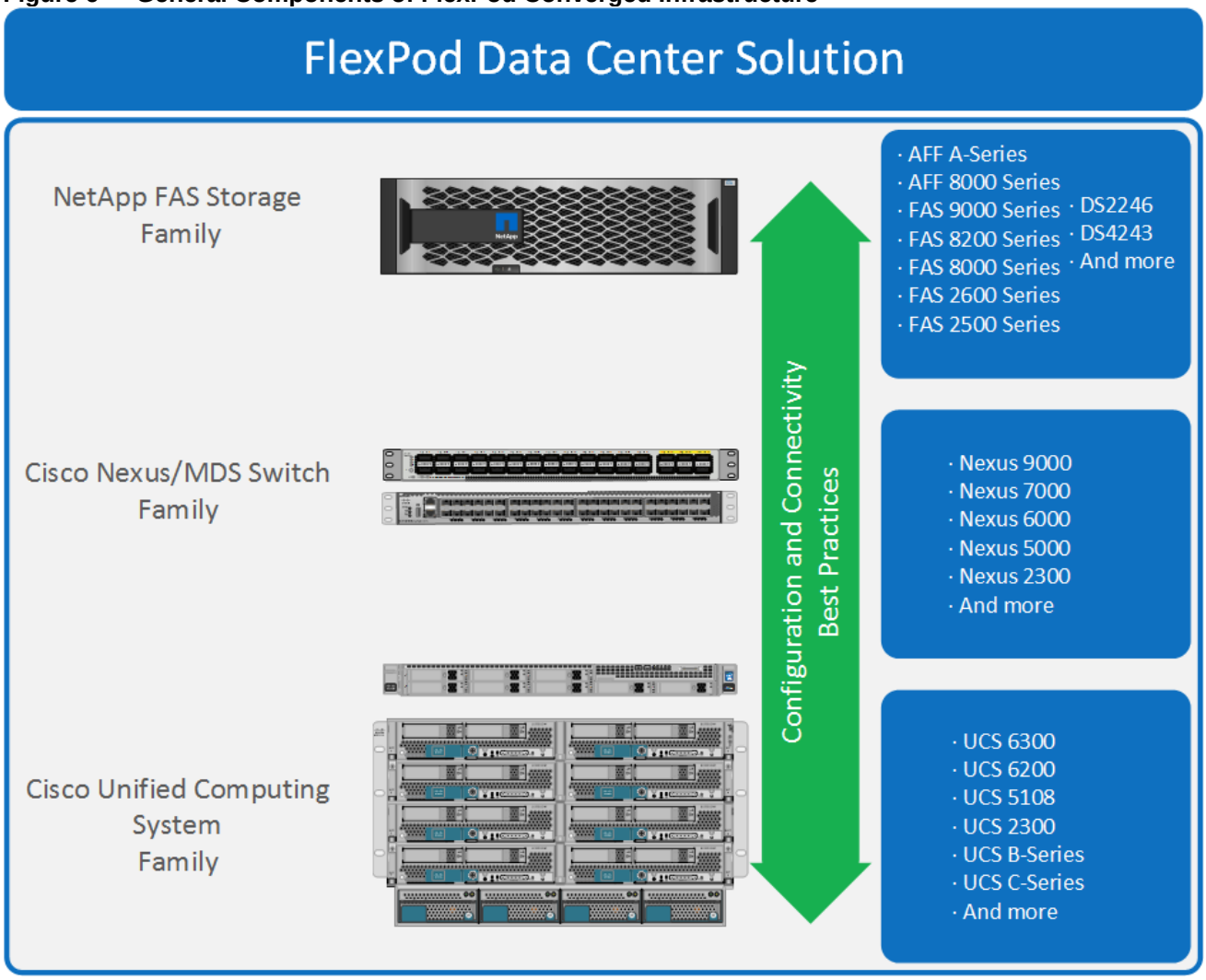
FlexPod Datacenter for IBM Cloud Private includes a FlexPod converged infrastructure and add-on IBM Cloud Private Software components. The infrastructure architecture in this solution is flexible and offers you various choices. Any supported FlexPod architecture can be used as the infrastructure platform to support IBM Cloud Private. The compute, network, and storage components are connected and configured according to the best practices of both Cisco and NetApp. These components are optimized for the deployment, operation and management of IBM Cloud Private as a highly scalable and flexible platform.

The reference architecture detailed in this document includes both design and deployment guides for the combined solution.

FlexPod Converged Infrastructure Components

The main components of the FlexPod Datacenter are introduced in this section followed by the add-on IBM Cloud private software components. For detailed information about these components and the design of the FlexPod Datacenter for IBM Cloud Private, please refer to the Solution Design and Solution Deployment sections.

Figure 6 General Components of FlexPod Converged Infrastructure



These components are connected and configured according to the best practices of both Cisco and NetApp to provide an ideal platform for running a variety of enterprise workloads with confidence. FlexPod can scale up for greater performance and capacity (adding compute, network, or storage resources individually as needed), or it can scale out for environments that require multiple consistent deployments (such as rolling out of additional FlexPod stacks).

One of the key benefits of FlexPod is its ability to maintain consistency during scale. Each of the component families shown (Cisco UCS, Cisco Nexus, and NetApp AFF) offers platform and resource options to scale the infrastructure up or down, while supporting the same features and functionality that are required under the configuration and connectivity best practices of FlexPod.

The following sections provide a technical overview of the compute, network, storage and management components of the FlexPod solution.

Cisco Unified Computing System

The Cisco Unified Computing System (Cisco UCS) is a next-generation data center platform that integrates computing, networking, storage access, and virtualization resources into a cohesive system designed to reduce total cost of ownership (TCO) and to increase business agility. The system integrates a low-latency; lossless unified network fabric with enterprise-class, x86-architecture servers. The system is an integrated, scalable, multi-chassis platform where all resources are managed through a unified management domain.

The Cisco Unified Computing System in the FlexPod architecture utilizes the following components:

- **Cisco UCS Manager (UCSM)** provides unified management of all software and hardware components in the Cisco UCS to manage servers, networking, and storage configurations. The system uniquely integrates all the system components, enabling the entire solution to be managed as a single entity through Cisco UCSM software. Customers can interface with Cisco UCSM through an intuitive graphical user interface (GUI), a command-line interface (CLI), and a robust application-programming interface (API) to manage all system configuration and operations.
- **Cisco UCS 6300 Series Fabric Interconnects** are a core part of Cisco UCS, providing both network connectivity and management capabilities for the system. The Cisco UCS 6300 Series offers line-rate, low-latency, lossless 10 and 40 Gigabit Ethernet, Fibre Channel over Ethernet (FCoE), and Fibre Channel functions. The unified fabric lowers costs by reducing the number of network adapters, switches, and cables, and by decreasing the power and cooling requirements.
- **Cisco UCS 5108 Blade Server Chassis** supports up to eight blade servers and up to two fabric extenders in a six-rack unit (RU) enclosure.
- **Cisco UCS B-Series Blade Servers** provide performance, efficiency, versatility and productivity with the latest Intel based processors.
- **Cisco UCS C-Series Rack Mount Servers** deliver unified computing innovations and benefits to rack servers with performance and density to support a wide range of workloads.
- **Cisco UCS Network Adapters** provide wire-once architecture and offer a range of options to converge the fabric, optimize virtualization and simplify management.

For detailed information about the Cisco Unified Computing System product line, see:
http://www.cisco.com/c/en/us/products/servers-unified-computing/index.html

Figure 7 Cisco Unified Computing System – Components

Cisco UCS 6332-16UP Fabric Interconnect

Cisco UCS 2304 Fabric Extender

Cisco UCS 5108 Blade Server Chassis

Cisco UCS B200 M5 Blade Server

Cisco Nexus 9332PQ

The Cisco Nexus 9300 platform consists of fixed-port switches designed for top-of-rack (ToR) and middle-of-row (MoR) deployment in data centers that support enterprise applications, service provider hosting, and cloud computing environments. They are Layer 2 and 3 non-blocking 10 and 40 Gigabit Ethernet switches with up to 2.56 terabits per second (Tbps) of internal bandwidth.

The Cisco Nexus 9332PQ Switch is a 1-rack-unit (1RU) switch that supports 2.56 Tbps of bandwidth and over 720 million packets per second (mpps) across thirty-two 40-Gbps Enhanced QSFP+ ports

All the Cisco Nexus 9300 platform switches use dual- core 2.5-GHz x86 CPUs with 64-GB solid-state disk (SSD) drives and 16 GB of memory for enhanced network performance.

With the Cisco Nexus 9000 Series, organizations can quickly and easily upgrade existing data centers to carry 40 Gigabit Ethernet to the aggregation layer or to the spine (in a leaf-and-spine configuration) through advanced and cost-effective optics that enable the use of existing 10 Gigabit Ethernet fiber (a pair of multimode fiber strands).

Cisco provides two modes of operation for the Cisco Nexus 9000 Series. Organizations can use Cisco® NX-OS Software to deploy the Cisco Nexus 9000 Series in standard Cisco Nexus switch environments. Organizations also can use a hardware infrastructure that is ready to support Cisco Application Centric Infrastructure (Cisco ACI™) to take full advantage of an automated, policy-based, systems management approach.

Figure 8 Cisco Nexus 9332PQ Switch

For detailed information about the Cisco Nexus 9000 product line, see:
http://www.cisco.com/c/en/us/products/switches/nexus-9000-series-switches/models-listing.html

NetApp Storage Controllers

NetApp AFF storage is included in this reference architecture as part of the FlexPod Datacenter for ICP. The AFF family addresses enterprise storage requirements with high performance, superior flexibility, and best-in-class data management, including features that accelerate DevOps. Built on ONTAP software, AFF speeds up the operations required to meet your business requirements, without compromising efficiency or reliability, while providing great flexibility and scalability. As true enterprise-class, all-flash arrays, these systems accelerate, manage, and protect business-critical data, now and in the future.

ONTAP 9.3 is used with the AFF A300 storage platform in our design. ONTAP data management software offers unified storage for applications that read and write data over block or file-access protocols in storage configurations that range from high-speed flash to lower-priced spinning media or cloud-based object storage.

ONTAP implementations can run on NetApp-engineered fabric-attached storage (FAS) or all-flash FAS appliances, on commodity hardware (ONTAP Select), and in private, public, or hybrid clouds (NetApp Private Storage, Cloud Volumes ONTAP, and ONTAP Select in the IBM Cloud). Specialized implementations offer best-in-class converged infrastructure as featured here as part of the FlexPod Datacenter solution and access to third-party storage arrays (FlexArray virtualization).

Together these implementations form the basic framework of the NetApp Data Fabric, with a common software-defined approach to data management and fast, efficient replication across platforms. FlexPod and ONTAP can serve as the foundation for both hybrid cloud and private cloud designs

NetApp Storage Virtual Machine (SVM)

A NetApp ONTAP cluster serves data through at least one and possibly multiple storage virtual machines (SVMs; formerly called Vservers). An SVM is a logical abstraction that represents the set of physical resources of the cluster. Data volumes and network logical interfaces (LIFs) are created and assigned to an SVM and might reside on any node in the cluster to which the SVM has been given access. An SVM might own resources on multiple nodes concurrently, and those resources can be moved non-disruptively from one node in the storage cluster to another. For example, a NetApp FlexVol® flexible volume can be non-disruptively moved to a new node and aggregate, or a data LIF can be transparently reassigned to a different physical network port. The SVM abstracts the cluster hardware, and thus it is not tied to any specific physical hardware.

An SVM can support multiple data protocols concurrently. Volumes within the SVM can be joined together to form a single NAS namespace, which makes all of an SVM's data available through a single share or mount point to NFS and CIFS clients. SVMs also support block-based protocols, and LUNs can be created and exported by using iSCSI, FC, or FCoE. Any or all of these data protocols can be configured for use within a given SVM. Storage admins and management roles can also be associated with SVM, which enables higher security and access control, particularly in environments with more than one SVM, when the storage is configured to provide services to different groups or set of workloads.

Storage Efficiencies

Storage efficiency has always been a primary architectural design point of ONTAP. A wide array of features allows you to store more data using less space. In addition to deduplication and compression, you can store your data more efficiently by using features such as unified storage, multitenancy, thin provisioning, and utilize NetApp Snapshot technology.

Starting with ONTAP 9, NetApp guarantees that the use of NetApp storage efficiency technologies on AFF systems reduce the total logical capacity used to store customer data by 75 percent, a data reduction ratio of 4:1. This space reduction is enabled by a combination of several different technologies, including deduplication, compression, and compaction.

Compaction, which was introduced in ONTAP 9, is the latest patented storage efficiency technology released by NetApp. In the ONTAP WAFL file system, all I/O takes up 4KB of space, even if it does not actually require 4KB of data. Compaction combines multiple blocks that are not using their full 4KB of space together into one block. This one block can be more efficiently stored on the disk to save space.

Encryption

Data security continues to be an important consideration for customers purchasing storage systems. NetApp supported self-encrypting drives in storage clusters prior to ONTAP 9. However, in ONTAP 9, the encryption capabilities of ONTAP are extended by adding an Onboard Key Manager (OKM). The OKM generates and stores keys for each of the drives in ONTAP, allowing ONTAP to provide all functionality required for encryption out of the box. Through this functionality, sensitive data stored on disk is secure and can only be accessed by ONTAP.

With ONTAP 9.1, NetApp has extended the encryption capabilities further with NetApp Volume Encryption (NVE), a software-based mechanism for encrypting data. It allows a user to encrypt data at the per-volume level instead of requiring encryption of all data in the cluster, thereby providing more flexibility and granularity to the ONTAP administrators. This encryption extends to Snapshot copies and NetApp FlexClone® volumes that are created in the cluster. One benefit of NVE is that it executes after the implementation of the storage efficiency features, and, therefore, it does not interfere with the ability of ONTAP to create space savings.

For more information about encryption in ONTAP, see:
https://library.netapp.com/ecm/ecm_download_file/ECMLP2572742

NetApp AFF A300

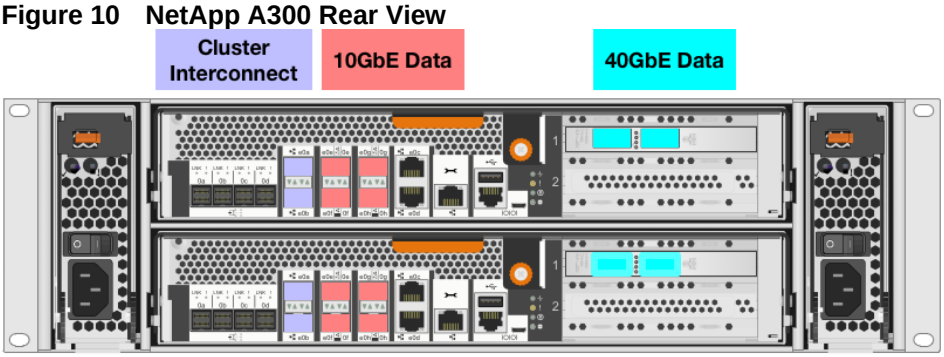
This design leverages NetApp AFF A300 controllers deployed with ONTAP 9.3. Although FlexPod supports a variety of NetApp FAS and AFF storage controller models, this CVD is based on AFF, an all flash-based SSD technology, that offers the best combination of performance, flexibility, and scalability.

The AFF A300 controller (see Figure 9 and Figure 10) provides the high-performance benefits of 40GbE and all-flash SSDs, This controller offers better performance than previous models in the midrange of the AFF family (for example, the AFF8040), while taking up only 3U of rack space for a two-node configuration (an HA pair in a single chassis). Combined with large capacity SSD disks, this solution can provide up to 11.7PB of raw capacity, based on the current limitation of 384 x 30TB SSDs. This makes it an ideal controller for a shared workload converged infrastructure. If you need more performance, then you can include higher end controllers than the AFF A300 in the FlexPod solution.

The 40GbE cards are installed in the expansion slot 2 and the ports are e2a, e2e.

Figure 9 NetApp A300 Front View

Figure 10 NetApp A300 Rear View



ONTAP provides included and optional software components. Table 1 lists the features and the capabilities of the NetApp AFF A300:

Table 1 NetApp AFF A300 Features and Capabilities

Bundle	Features and Software	Included with A300
ONTAP Base License	• Efficiency: FlexVol®, inline deduplication, inline compression, inline compaction, and thin provisioning • Availability: Active-Active HA pair and Multipath I/O • Data protection: RAID-DP®, RAID TEC, and Snapshot® • Synchronous replication for disaster recovery: MetroCluster™ • Performance Control: Adaptive quality of service (QoS), balanced placement	Yes

Bundle	Features and Software	Included with A300
	• Management: OnCommand® Workflow Automation, System Manager, and Unified Manager • Scalable NAS container: FlexGroup	
NetApp Flash Bundle	• All storage protocols supported (FC, FCoE, iSCSI, NFS, pNFS, SMB) • SnapRestore®: Backup and restore entire Snapshot copies in seconds • SnapMirror®: Simple, flexible replication for backup and disaster recovery • FlexClone®: Instant virtual copies of files, LUNs, and volumes • SnapCenter®: Unified, scalable platform and plug-in suite for application-consistent data protection and clone management • SnapManager®: Application-consistent data backup and recovery for enterprise applications	Yes
Extended Value Software	• NVMe™ over Fibre Channel (NVMe/FC) protocol: faster and more efficient host connection than original Fibre Channel • OnCommand Insight: Flexible, efficient resource management for heterogeneous environments • SnapLock®: Compliance software for write once, read many (WORM) protected data • Volume Encryption (free license): Granular, volume-level, data-at-rest encryption • FabricPool: Automatic data tiering to the cloud	Optional

For more information about the AFF A-series product family, see: http://www.netapp.com/us/products/storage-systems/all-flash-array/aff-a-series.aspx.

Disk Shelves

Two DS224C disk enclosures are attached to the controllers. These enclosures are each populated with 24 x 900GB SSDs (48 total). Higher capacity media is available and supported, including 3.8TB SSDs and 15.3TB SSDs. The DS224C also supports NVMe drive packs.

Figure 11 DS224C Disk Enclosure

Alternative disk enclosures are available if there is a need to support hybrid configurations of SSDs and NL-SATA drives.

For more information about the optional disk enclosures, see:

https://www.netapp.com/us/products/storage-systems/disk-shelves-and-storage-media/index.aspx#tech-specs

For a comprehensive list of storage hardware specifications, including supported media type and capacities, see the NetApp Hardware Universe.

VMware vCenter Server

VMware vCenter is a centralized management platform for VMware environments. It provides visibility, scalability and automation to the virtual infrastructure. Its key features include the following:

- Simple deployment and administration
- Centralized control and visibility
- Scalability and extensibility across hybrid cloud
- Plug-in extensibility
- Native backup and restore capabilities
- Management of 1000 hosts and up to 10,000 VMs with a single vCenter instance

This document assumes that vCenter is already installed in your environment. Therefore, the ESXi servers that host the ICP nodes, as well as possibly additional enterprise applications, are managed from the existing vCenter and VMware environment. Installing and configuring vCenter is beyond the scope of this document.

For more information, see: http://www.vmware.com/products/vcenter-server/overview.html

FlexPod Datacenter for IBM Cloud Private Add-on Components

The following subsections describe the add-on components that are part of the solution along with the core FlexPod Datacenter converged infrastructure and the vSphere hypervisor. The add-on components are:

- IBM Cloud Private software (version 2.1.02)
- Enterprise Bundle for IBM Cloud Private
- NetApp Trident, a dynamic storage provisioner for Kubernetes

IBM Cloud Private 2.1.02

IBM Cloud Private is an application platform for developing and managing on-premises, containerized applications. It is an integrated environment that includes Kubernetes as the containers orchestration platform, a private image repository, management console, and monitoring framework. It also contains tools for integration with public cloud services that minimize risk associated with deployment of new applications.

In this reference architecture, we used, tested, and validated ICP version 2.1.02 and its out-of-the-box features and packages. An additional set of software components that are available with the Enterprise bundle are not packed with ICP. Customers that purchase the Enterprise edition have the flexibility to install the various software products in combination of containers, which then run in the ICP platform itself, or as a VM. These products can also be installed in a more traditional way as set of VMs running on the ESXi servers or directly on the UCS servers in a bare metal type deployment.

The following list of ICP use cases is based on product positioning from IBM and the challenges the solution can solve for customers. Also listed are the key features that enable these use cases.

The leading ICP Use Cases:

1. Cloud Native Application Development
   Streamline development with built-in micro-services, runtimes, containers and Kubernetes orchestration plus integrated management.

2. Application Modernization
   Move your apps as-is to the cloud or re-factor them for use in development and application workload models

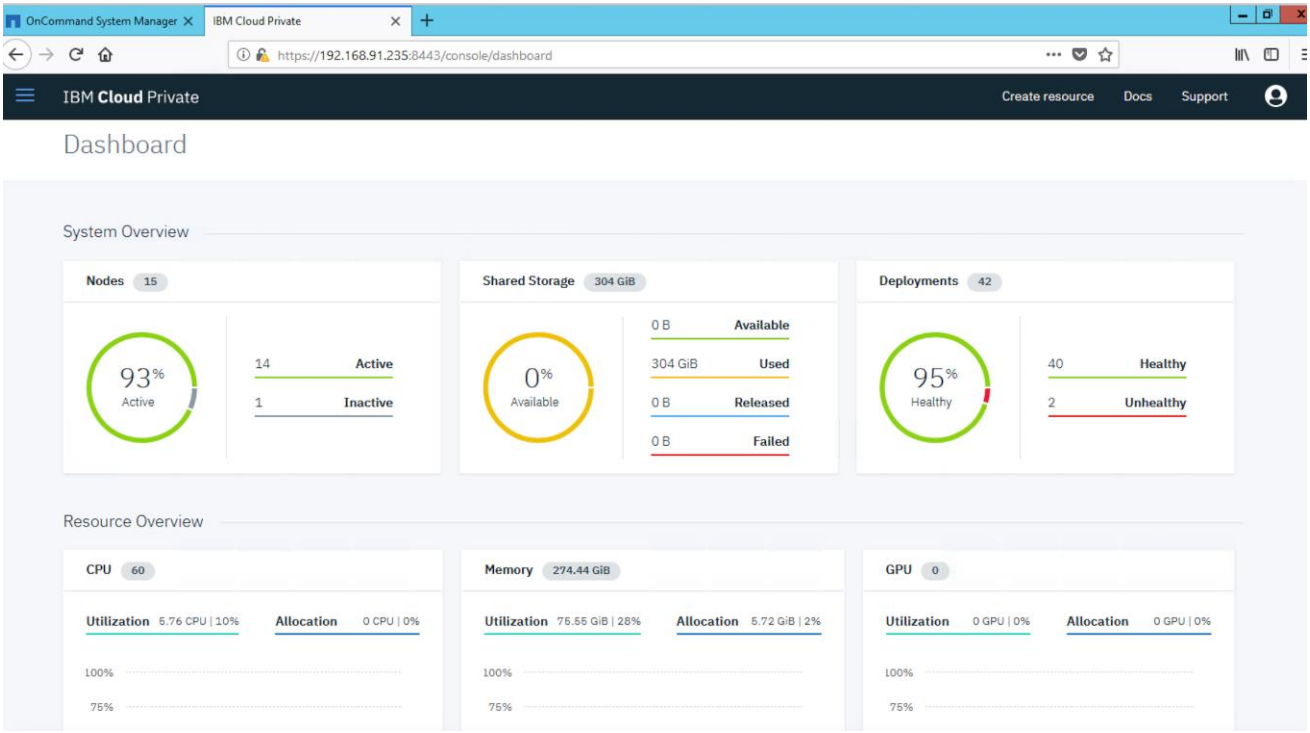
3. Open up enterprise data center to work with cloud services
   Leverage your existing applications and data in a security-rich environment, while developing innovative applications and services.

For more information about the use cases, see: https://www.ibm.com/cloud/private

To accelerate adoption and productivity, ICP includes the following features and components:

- **Built on Kubernetes and Docker**. Both Kubernetes and Docker are considered by the market to be the go-to standards for containers and container management technologies.
- **Open Source.** Out-of-the-box ICP includes many open source products and tools that are widely adopted by the market, specifically in the developer's community. Some examples are Jenkins for CI/CD pipelines and the Node.JS JavaScript run time environment. Management tools for monitoring, alerting, logging and metering such as Grafana, Prometheus, Kibana and more are also based on open source projects and included with ICP.
- **Private image repository and catalog.** A private image repository is available for customers to store their own set of Docker images which can then be easily accessible through the catalog. The catalog can be populated with dozens of containerized applications ready to be deployed, including Jenkins, WebSphere Liberty, Db2 warehouse, Ngnix, MongoDB, various IBM tools and more. You can push or pull images from a local file system or shared repository to the private image registry in ICP.
- **GUI for operations and management**. Although many people interact with the cluster by using the kubectl CLI, the underlying Kubernetes GUI used by ICP is simple, easy to navigate, and, along with the content already populated by IBM in the catalog and the additional tools, helps streamline the operation of regular daily tasks.

Figure 12 ICP Dashboard

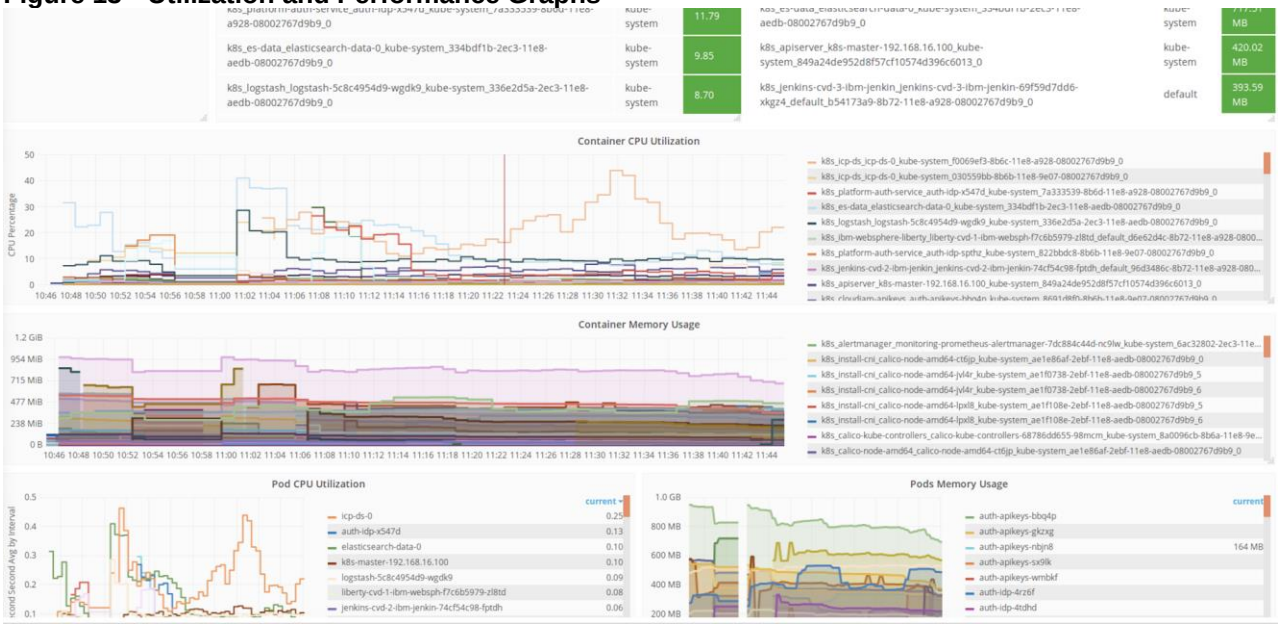


- **Logging.** IBM Cloud Private uses the ELK stack to collect system logs, including Kubernetes and Docker. ELK stacks can be deployed from the catalog to different namespaces in the environment and can be configured to collect Docker applications log as well.

  For more information about logging in ICP, see: https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/manage_metrics/logging_elk.html

- **Monitoring and alerting.** ICP uses Prometheus and Grafana to monitor the status of applications and the cluster. Prometheus collects data, and Grafana provides the visualization. Prometheus can also be configured for alerts, although you can also use external alerting tools.

Figure 13 Utilization and Performance Graphs

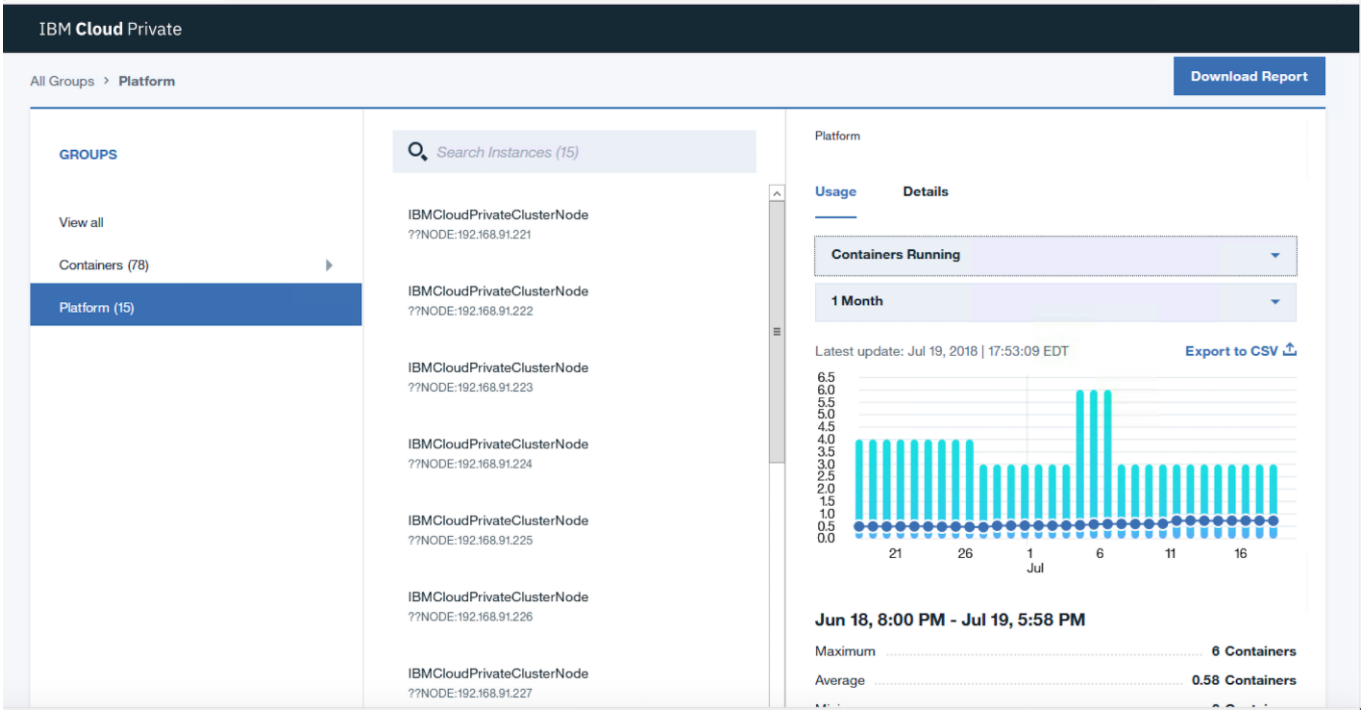


For more information about monitoring and alerting, see:
https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0/manage_metrics/monitoring_service.html

Alerts rules can also be set and configured to send notifications.

- **Metering.** Metering services are automatically installed in ICP and can be used to view and download the usage metrics for applications and the cluster.

Figure 14 Example of Metering Report



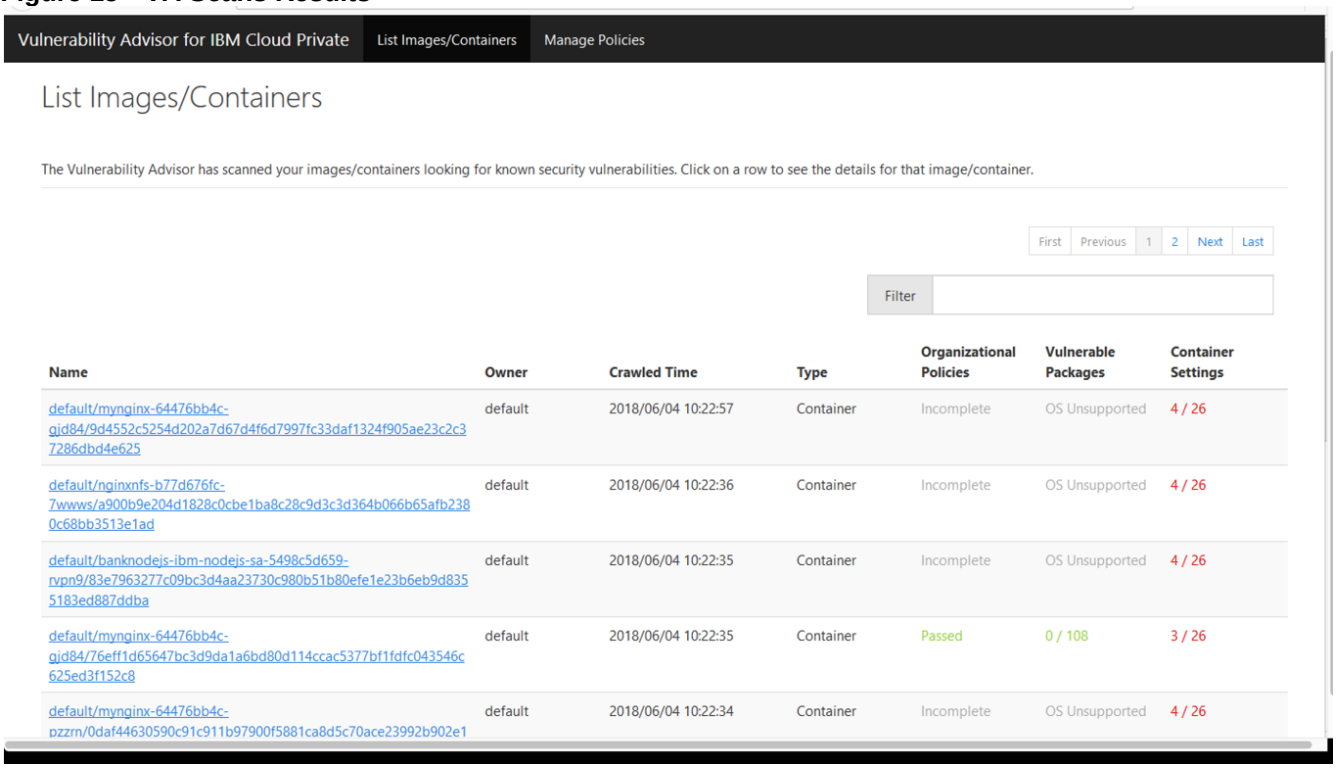
For more information about ICP metering services, see:
https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/manage_metrics/metering_service.html

- **Roll-based access control.** IBM Cloud Private supports several roles based on the available actions. RBAC for the Catalog and Helm resources as well as for the Kubernetes resources are based on IAM (Identity & Access Management) roles, and ICP supports the following IAM roles: viewer, operator, editor, administrator, and cluster administrator.

  For more details about ICP and RBAC, see: https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0.2/user_management/assign_role.html

- **Vulnerability Advisor.** VA is included in ICP (from version 2.1.02) to perform security checks on images before deployment and while containerized applications are running.

Figure 15 VA Scans Results

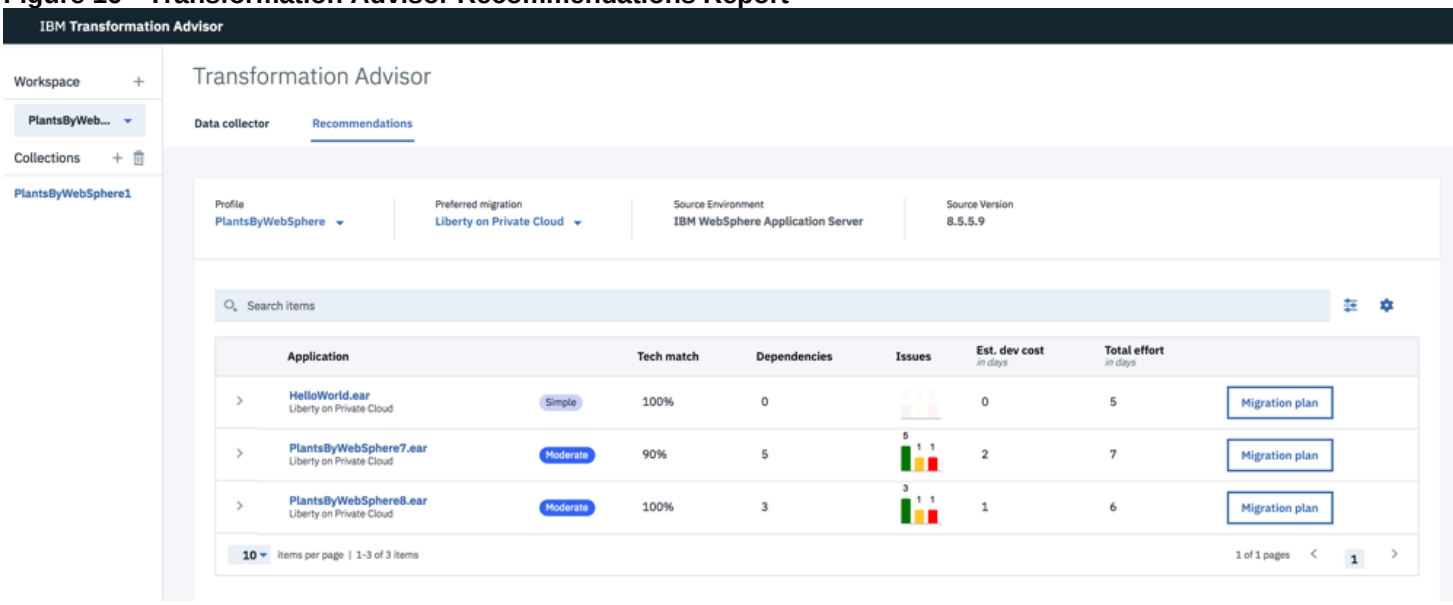


For more information about Vulnerability Advisor, see: https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0.2/manage_cluster/vuln_advisor.html

- Transformation Advisor (TA). TA can be deployed from ICP to support application modernization. It enables you to evaluate on-premises JAVA EE applications for rapid deployment on a WebSphere application server, on Liberty running on ICP, and also in public cloud environments. This tool identifies potential issues and gaps for moving applications to the cloud and provides a plan to transform a system to a containerized cloud-native solution.

Figure 16 provides an example of report with issues and a proposed migration plan in TA.

Figure 16 Transformation Advisor Recommendations Report



For more information about Transformation Advisor, see: https://developer.ibm.com/product-insights/transformation-advisor/

ICP – Enterprise Bundle

The enterprise bundle for ICP is an add-on set of IBM software packages listed and described earlier in the Application Modernization and Enterprise Application Use Cases section of the document. These software packages are not part of the ICP installation package itself, but once obtained, usually from the IBM Passport Advantage site, they can be deployed in the ICP platform as Helm charts, so as a containerized version. Based on use cases, needs and versions, customers may deploy the enterprise applications also as a VM or on bare metal servers, knowing that FlexPod can support all deployment models.

NetApp Trident – Dynamic Storage Provisioner for Kubernetes

Trident is an open source project created by NetApp for the Kubernetes community. Trident has been implemented as an external provisioning controller that runs as a pod. It monitors Kubernetes volumes and completely automates the provisioning process. The following common use cases can take advantage of persistent storage support for Kubernetes:

- DevOps teams who want to accelerate the CI/CD pipeline
- Traditional enterprise applications deployed into the hybrid-cloud
- Cloud native applications and microservices

In addition to persistent volume integration, Trident also provides the following advanced capabilities that provide storage deployment flexibility for containerized applications:

- Configuration with a simple REST API by using unique abstractions that tie specific capabilities to Kubernetes storage classes.
- Application data managed and protected by enterprise-class storage. Existing storage objects, such as volumes and LUNs, can be easily consumed by Trident.
- Use of multiple storage back-ends. Deploying each back-end with a different configuration allows Trident to provision and consume storage with different characteristics and costs. Trident can present the storage infrastructure to container deployed workloads in a straightforward fashion, without complexity.

There is a wide variety of use cases for persistent storage with containers, from monolithic applications to 12-factor microservices—most DevOps workflows can benefit from development and deployment in containers. Workloads that require stateful data with containers include databases, continuous integration/continuous deployment, big data, and many more. No matter your use case or workload, the Trident can enable a faster, more agile software lifecycle.

Containers abstract the application from the underlying operating system, enabling portability and flexibility in software development. And the introduction of persistent storage across various container technologies is essential for data mobility. By pairing Kubernetes with Trident, the FlexPod solution brings you closer to a cloud-native future.

Because IBM Cloud Private is based on Kubernetes, you can benefit from Trident as the dynamic storage provisioner. Therefore it is an included component of the solution. Trident is deployed as a pod and simplifies the provisioning of persistent storage within the Kubernetes framework. With Trident, customers can perform the following tasks:

- Improve data protection with underlying storage Snapshot copies.
- Accelerate CI/CD workflows by leveraging the underlying cloning capabilities of the ONTAP storage platform (FlexClone).
- Replicate data for disaster recovery and other purposes by enabling data protection policies on the back-end storage platform (and utilizing SnapMirror).

Various storage services can be configured at the storage level. They are presented to the Kubernetes layer as different storage classes to be consumed based on different attributes such as performance, data protection policies, cost, and other aspects to meet business requirements and operational practices.

Trident provisions storage services (Persistent Volumes – PVs, and Persistent Volume Claims – PVCs) requested by the end-user of the ICP platform, which are typically developers. The storage administrator is not involved in the process of provisioning storage. The role of the storage administrator and architects is to design the back-end storage services in a way that they can be consumed by the ICP platform with Trident.

These design aspects include the following considerations:

- Setting up the storage capacity and performance features
- Establishing data protection policies
- Setting encryption options
- Creating SVMs and access policies

Details about the considerations regarding setting up the storage for ICP are covered in the design and deployment sections of this document.

Trident is deployed as a pod with two containers (trident-main and etcd). Trident includes nDVP (NetApp Docker Volume Plugin) as one of its library to interact with the Docker containers.

Prior to interacting with the back-end storage (an AFF A300 in our case), storage drivers must be defined in the cluster. Trident supports the following drivers for ONTAP storage systems; AFF, FAS, ONTAP Select and Cloud ONTAP (Table 2).

Table 2 Trident Drivers

Driver	Protocol
ontap-nas	NFS
ontap-nas-economy	NFS
ontap-san	iSCSI

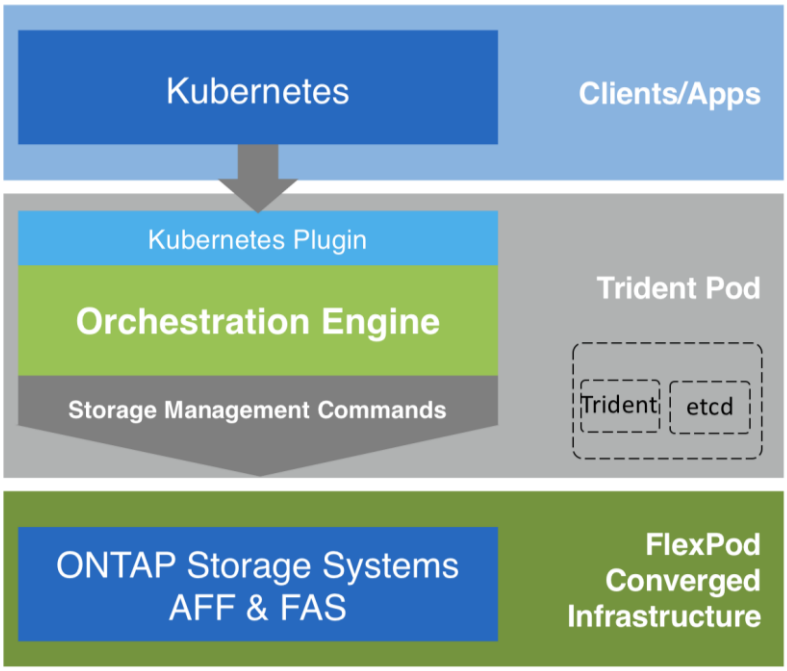
The ontap-nas and ontap-san drivers map each persistent volume to a FlexVol entity in the back-end storage. ONTAP supports up to 1000 FlexVol volumes per storage cluster node and up to 12,000 FlexVol volumes total in a multi-node cluster. These drivers are preferred due to the granular data management capabilities associated with FlexVol volumes.

If larger number of instances is required, the ontap-nas-economy driver is available to support up to 100,000 volumes in a node cluster and up to 2.4 million volumes in a multi-node cluster. The ontap-nas-economy driver uses qtrees to represent volumes to the Kubernetes cluster.

You can configure more than one driver with different associated storage services or classes. For example, a gold storage class can be configured to use the ontap-nas driver with a particular data protection policy and high performance SSD media. Alternatively, a bronze class can be configured to use the ontap-nas-economy driver with no data protection and lower cost and lower performance media such as NL-SAS.

The architecture of Trident is illustrated in Figure 17.

Figure 17 Trident Architecture



For more information about Trident, see: https://netapp.io/2016/12/23/introducing-trident-dynamic-persistent-volume-provisioner-kubernetes/

Solution Design

This section provides an overview of the hardware and software components used in this solution, as well as the design factors to be considered in order to make the system work as a single, highly available solution.

Architectural Overview

The FlexPod Datacenter for Private Cloud with IBM Cloud Private provides an end-to-end architecture with Cisco and NetApp technologies that demonstrate support for IBM Cloud Private workloads with high availability and server redundancy. The architecture consists of IBM Cloud Private Software components deployed on FlexPod architecture with Cisco UCS B-Series blade servers and NetApp AFF A300 series storage attached to the Cisco Nexus 9332PQ Nexus standalone switches. This infrastructure provides iSCSI and FC boot options for hosts with file-level and block-level access to shared storage.

The IBM cloud private in this solution has been deployed on Ubuntu virtual machines running on FlexPod that consists of VMware vSphere 6.5 u1 hypervisor, installed on Cisco UCS M5 servers. The Cisco UCS M5 blade servers are connected over 40 Gbps Ethernet to Cisco UCS 6332 series fabric interconnects. The NetApp A300 is connected to the Nexus based network fabric over 40Gbps Ethernet. The ESXi nodes supporting IBM cloud private environment and the ICP worker nodes access internet Small Computer System Interface (iSCSI) volumes for the boot and NFS file storage for data stores and the application persistent volumes respectively.

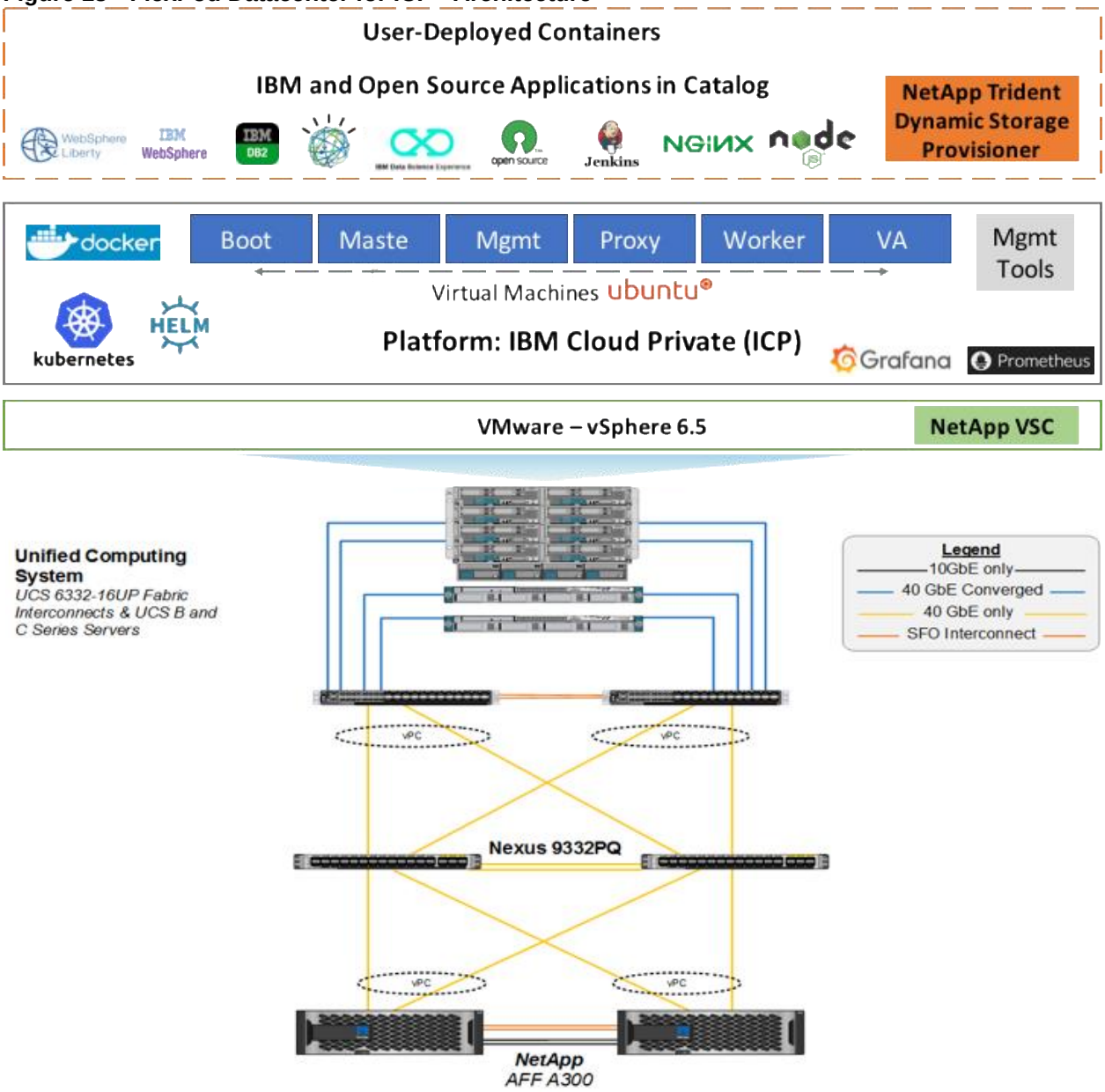
Figure 18 illustrates a base design. Each of the components can be scaled easily to support specific business requirements. For example, additional ICP nodes can be deployed to scale the ICP environment or even blade chassis can be deployed to increase compute capacity, additional storage controllers or disk shelves can be deployed to improve I/O capability and throughput, and special hardware or software features can be added to introduce new features.

The infrastructure services such as Active Directory, DNS, NTP and VMWare vCenter are deployed outside the FlexPod system. In case of customers have these services already available in their data center, these services can be leveraged to manage the FlexPod system with IBM Cloud Private.

This reference architecture is based on a highly available medium size ICP configuration. Refer to IBM's guidelines for ICP hardware:
https://www.ibm.com/support/knowledgecenter/en/SSBS6K_1.2.0/supported_system_config/hardware_reqs.html

Figure 18 provides a high-level overview of the FlexPod Datacenter for ICP architecture.

Figure 18 FlexPod Datacenter for ICP - Architecture



FlexPod Physical Topology

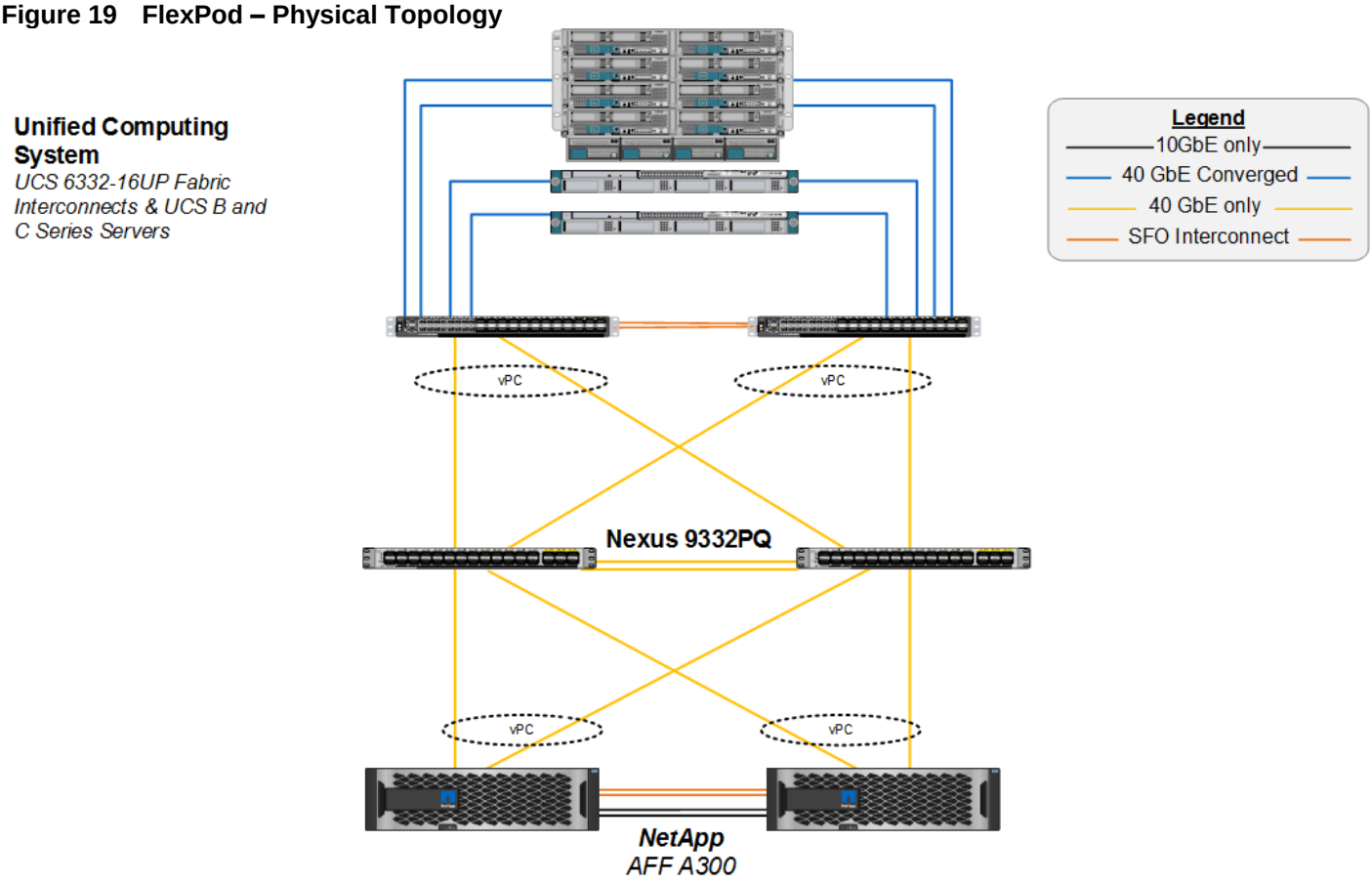
The Figure 19 shows the FlexPod Datacenter components and the network connections for a configuration with the Cisco UCS 6332-16UP Fabric Interconnects. This design has end-to-end 40 Gb Ethernet connections between the Cisco UCS 5108 Blade Chassis and the Cisco UCS Fabric Interconnect, between the Cisco UCS Fabric Interconnect and Cisco Nexus 9000, and between Cisco Nexus 9000 and NetApp AFF A300. This infrastructure is deployed to provide iSCSI-booted hosts with file-level and block-level access to shared storage. The reference architecture reinforces the "wire-once" strategy, because as additional storage is added to the architecture, no re-cabling is required from the hosts to the Cisco UCS fabric interconnect.

The FlexPod datacenter design is not discussed in detail but important elements to support FlexPod for IBM Cloud Private Solution are covered. For detailed information about the FlexPod architecture, refer to the following Design and Deployment guides:

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/flexpod_esxi65u1design.html

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/flexpod_esxi65_n9kiscsi.html

Figure 19 FlexPod – Physical Topology



FlexPod Logical Topology

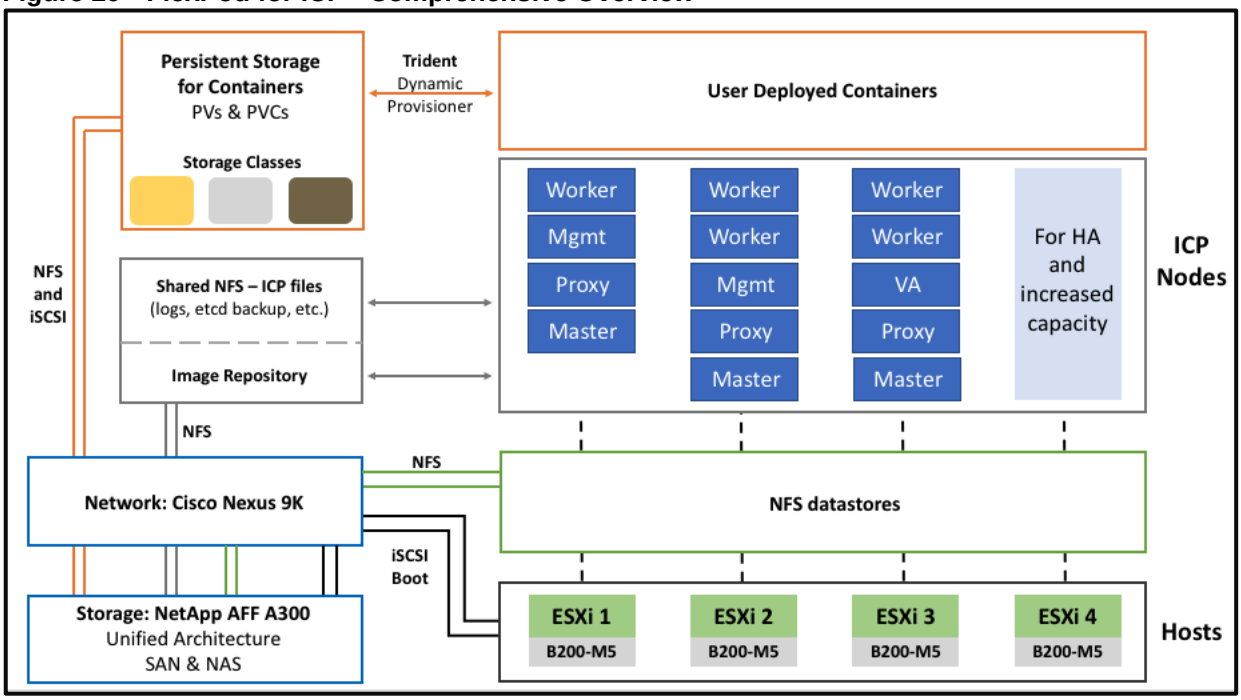
Figure 20 illustrates the FlexPod Datacenter with IBM Cloud Private logical topology with IBM cloud private components utilizing compute, network, and storage resources on FlexPod. The storage and network connectivity to IBM Cloud Private Cluster nodes running on VMware vSphere HA cluster (deployed on the Cisco UCS B200 M5 servers) is enabled by the Cisco Nexus 9k switches within FlexPod. The IBM Cloud Private environment has been deployed on FlexPod without any major modifications made to the infrastructure.

Persistent storage is a critical part of running stateful containers, and IBM Cloud Private with Kubernetes simplifies storage management by abstracting details of how storage is provisioned and how it is consumed. Persistent volumes for containers can be static or dynamically provisioned, in this case it is dynamic with FlexPod and is enabled by NetApp Trident. Dynamic volume provisioning allows storage volumes to be created on-demand, NetApp Trident for Containers eliminates the need to pre-provision storage for containers and allows persistent storage provisioning during the container deployment.

The storage resources can be dynamically provisioned using the Trident provisioner specified by the StorageClass object. StorageClasses are essentially blueprints that abstract away the underlying storage provider, as well as other parameters, like disk-type; ex: SSD vs HDD. This solution used NFS storage for dynamic storage

provisioning with optional support for iSCSI block storage, the ICP configuration files have also been stored on shared NFS file share for high-availability and to provide recovery capabilities for ICP environment.

Figure 20 FlexPod for ICP - Comprehensive Overview



FlexPod Storage Design for ICP

The FlexPod Datacenter for ICP is based on AFF A300 storage system with ONTAP version 9.3 (OS). A NetApp plug-in (called Trident) for Kubernetes is an add-on component that needs to be installed on a server with kubectl access to the Kubernetes cluster. Trident enables the integration between the storage and ICP, as ICP runs on a Kubernetes cluster.

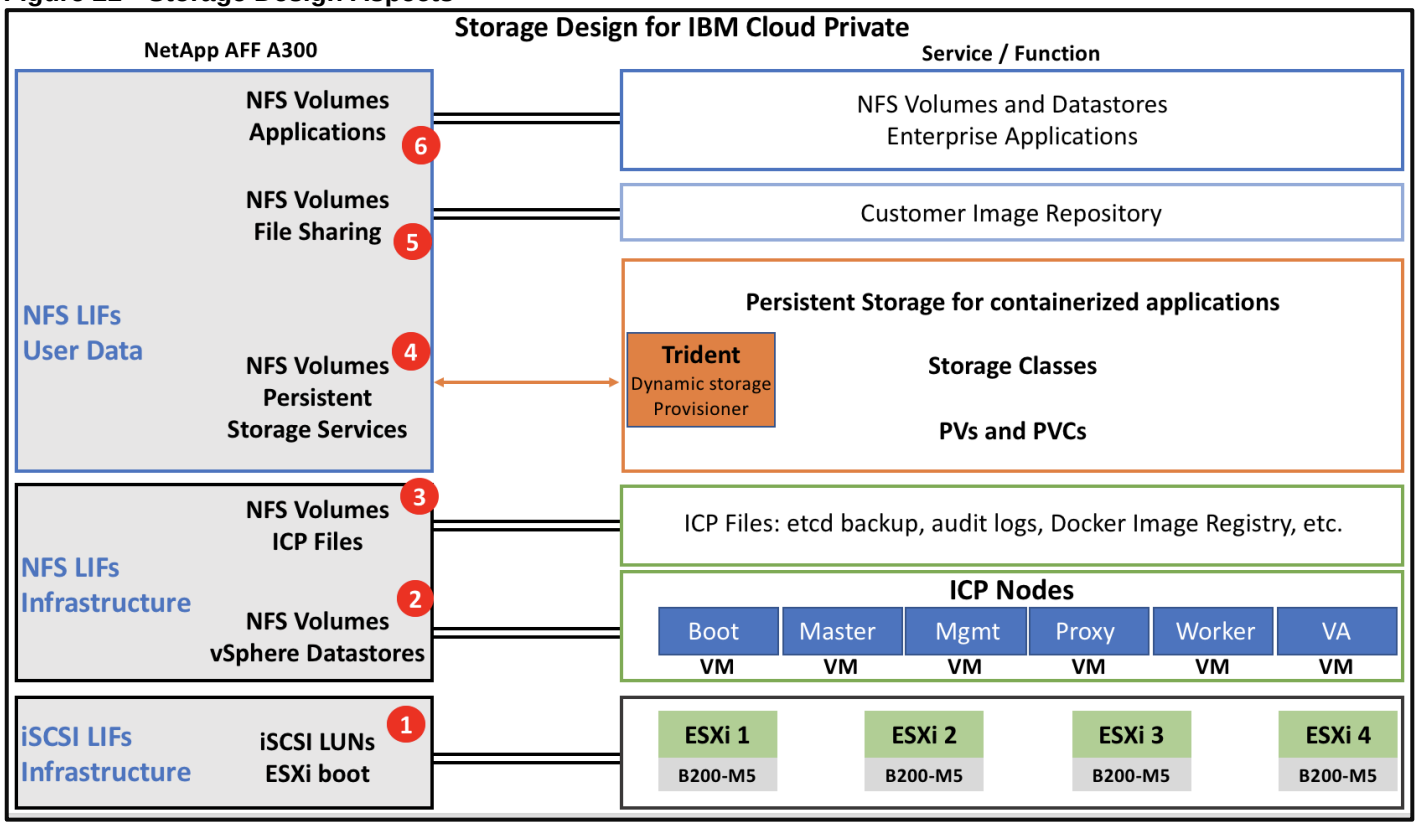
In addition to Trident, detailed in the Solution Deployment section, the A300 provides various storage services as illustrated in Figure 21.

The unified architecture of the storage is being used both as SAN and NAS for ICP. For the purpose of this validated design, iSCSI was used for the ESXi servers boot and NFS was used for the underlying datastores and for shared volumes used by various ICP files and additional applications that may be hosted on the FlexPod stack.

0 illustrates the various storage services of the solution while emphasizing the following unique attributes:

1. Provide a full range of storage services to ICP and its accompanying applications
2. Unified architecture that offers the flexibility of both SAN and NAS on the same storage device at the same time
3. Integration with Kubernetes via Trident for dynamic persistent storage provisioning
4. Integration with VMware
5. Fully redundant and highly available path from the storage controller via the network switches to the servers

Figure 21 Storage Design Aspects



iSCSI and NFS licenses should be added and enabled on the A300 storage platform prior to setting up iSCSI and NFS.

The A300 with ONTAP provides an unmatched feature-rich set of storage services for ICP. The list below corresponds to the different numbered storage services shown in Figure 21.

1. iSCSI boot
   iSCSI SAN is used for boot LUNs for the ESXi servers. Redundant iSCSI LIFS (Logical interfaces) are configured on each of the two storage controllers. The iSCSI infrastructure is associated with an infrastructure SVM (Infra-SVM)

2. Two NFS datastores are configured for the ICP nodes. The datastores are available to all ESXi servers. In the event of a storage controller failure, the HA pair takes over for continuous operation, making the solution is highly available. These datastores server the ICP nodes. For granularity of operation and optimization of storage functions to meet different services, such as data protection, storage efficiencies, etc. it is recommended to construct different datastores for the user containers. Additional details are provided in the deployment section.

3. Several NFS Volumes were created to store key ICP files. To serve NFS data and be accessed by the ICP nodes these volumes need to be exported and then mounted at the nodes. The Ubuntu nodes of ICP must have NFS services installed prior to mapping the NFS exports of the storage. The shared NFS volumes provide highly available storage hence improve the availability of the entire solution. Without highly available storage the files will be stored on the local node servers, which increases the risk due to failure but also complicate operations as some files need to be accessible to the entire ICP cluster so can't be stored on one particular server.
   To further improve the quality of the NFS services for ICP customers can enable Snapshots of these volumes and also enable the auto-grow capability to support dynamic growth as and if needed.
   Log files, repository of Docker images, and etcd backup are some of the key files that require shared storage.

4. Trident can access the ONTAP storage using NFS and iSCSI drivers. In this solution the team focused on NFS for simplicity but iSCSI can be used instead or in addition as both Trident and the A300 model support both protocols.
   Once Trident is installed and the backend storage is configured and the storage classes are defined, end users can deploy containers with persistent storage dynamically via the Kubernetes framework using PVs and PVCs and the appropriate storage class available for them.

5. A shared NFS volume can also host end-user image repository and ICP can be pointed to this location. This repository can be protected with the storage Snapshots and it can also be replicated to a DR site if needed using SnapMirror.
   Modifying images can be accelerated with FlexClone that can be operated at a file or volume level.

6. NFS volumes for Enterprise Applications. As established earlier in the document, additional storage services may be required to support Enterprise applications. Enterprise applications may be acquired as an ICP bundle (Enterprise bundle) and as one of the options, to be installed as part of the ICP environment and deployed as VMs on the ESXi servers within the FlexPod stack on NFS datastores. In addition to VMware datastores, certain data sets need to be stored on shared volumes so they can be accessible to multiple concurrent applications and services, and some data needs to be stored on highly available storage so it can be protected by snapshots or be replicated to a DR site.

   Across all storage services listed above, customers can also benefits from Quality of Services (QoS), Data Encryption, Storage Efficiencies (Thin provisioning, Deduplication, Compaction, Compression and Thin replication), protect the data with Snapshots and SnapRestore, Replicate data for DR purposes, and for enabling hybrid cloud deployment with SnapMirror and accelerate CI/CD and other DevOps processes by utilizing ONTAP's cloning feature (FlexClone).

iSCSI LIFs are used for SAN boot of the ESXi servers. This is a redundant path at the level of each of the two storage controllers, the switches, and the NICs on the servers.

The NFS LIFs for the infrastructure provide the NFS services for the VMware datastores as well as NFS volumes for ICP files that require shared storage that is highly available in order to minimize failures of the platform. By default, these files will be stored on the local node which is not protected at the same level as data stored on the A300. More details about protecting key ICP files are provided in the Solution Deployment section.

Best practices and design aspects are documented in previously published CVDs, including FlexPod Datacenter with VMware vSphere 6.5.

Please refer to NetApp TR-4067 and NetApp TR-4073 for best practices associated with NFS, including security; NetApp NFS Best Practices: https://www.netapp.com/us/media/tr-4067.pdf

NetApp Secure Unified Authentication Kerberos, NFSv4, and LDAP in ONTAP:
https://www.netapp.com/us/media/tr-4073.pdf

NFS export policies:
https://library.netapp.com/ecmdocs/ECMP1196891/html/GUID-9A2B6C3E-C86A-4125-B778-6072A3A19657.html=

Storage Services

To offer optimal cost/performance storage services that meet the requirements for deploying containers, it is recommended to design the backend storage with different storage services. Some containers will require high performance storage with data protection policies, for others, data protection will not be required and perhaps high performance is also not needed. Designing the right storage services for the organization requires collaboration between the development team that will be consuming the services and the storage and infrastructure team. It is common to design the storage with 1–3 types of distinct services but there's no limit at the ONTAP level and different combinations of the media type supported by the backend platform and the available features can be incorporated into a storage service.

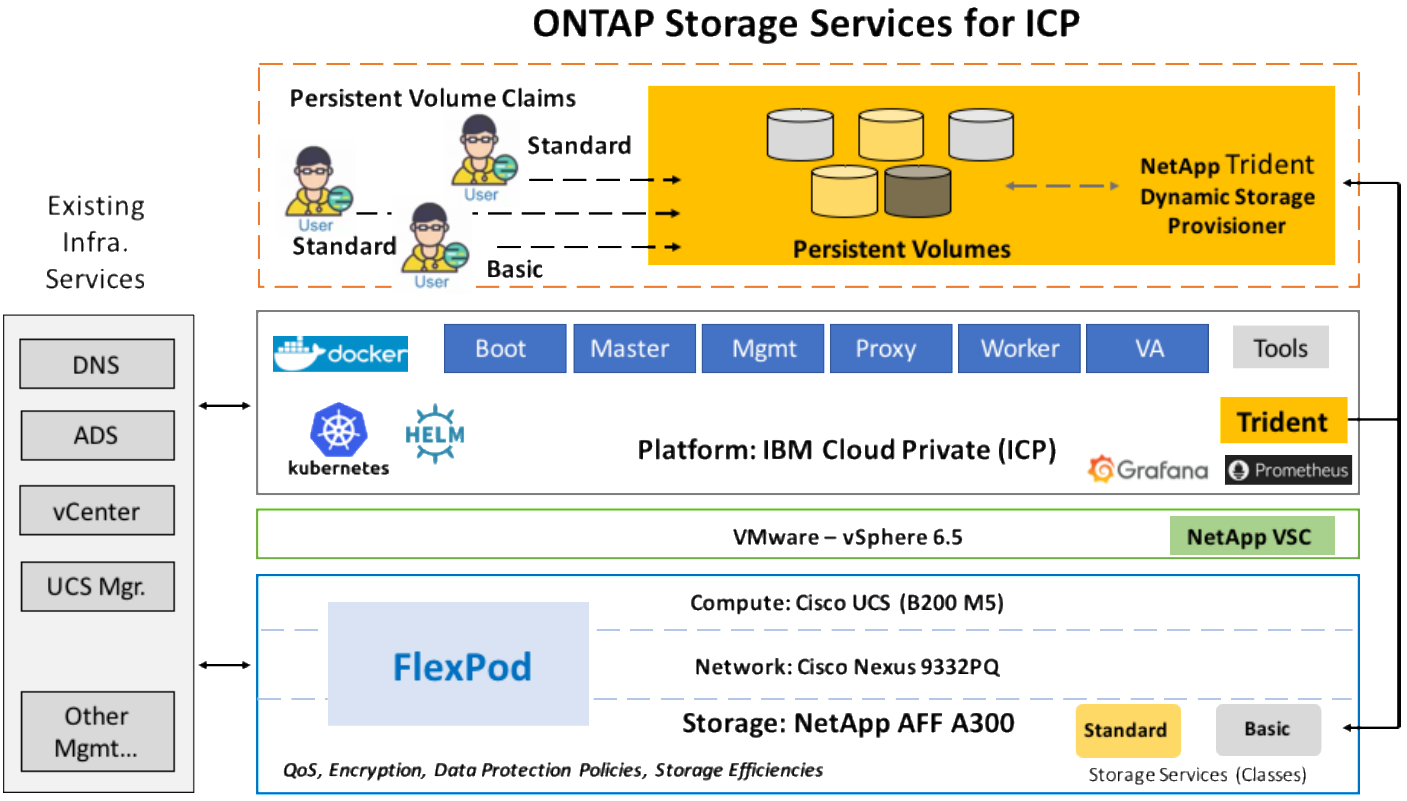
The following are the storage attributes that may be used when designing storage services:

- Media type: SSD , SAS, NL-SAS (as supported by the AFF or FAS storage controllers)
- Encryption
- Backup (Snapshot) policies
- Data Replication (SnapMirror) policies
- Storage efficiencies: Deduplication, compression, compaction
- Protocols (NFS, iSCSI, FC)
- Quality of Services (QoS)

These combination of storage configurations set by the storage admin can then be turned into services such as “Gold”, “Silver” and “Bronze” or “Basic” and “Standard” as used in our test environment. For the ICP platform, these storage services will be defined in .yaml files for Trident, the dynamic storage provisioner, to use as the storage classes for the persistent Volume Claims (PVC) requested by the users.

Figure 22 depicts how the storage services are being used by Trident from the Kubernetes/ICP platform.

Figure 22 Storage Services with Trident for ICP



Storage Considerations for Trident

Trident uses NFS export policies to control access to the volumes that it provisions. It uses the default export policy unless a different export policy name is specified in the configuration.

While Trident associates new volumes (or Qtrees) with the configured export policy, it does not create or otherwise manage export policies themselves. The export policy must exist before the storage backend is added to Trident, and it needs to be configured to allow access to every worker node in the Kubernetes cluster.

If the export policy is locked down to specific hosts, it will need to be updated when new nodes are added to the cluster, and that access should be removed when nodes are removed as well.

Trident uses etcd to maintain state for the objects that it manages. This is Trident's etcd and not the etcd included with Kubernetes.

By default, Trident deploys an etcd container as part of the Trident pod. This is a single node etcd cluster managed by Trident that is backed by a highly reliable volume from a NetApp storage system. This is perfectly acceptable for production given the redundancy of the ONTAP storage system deployed.

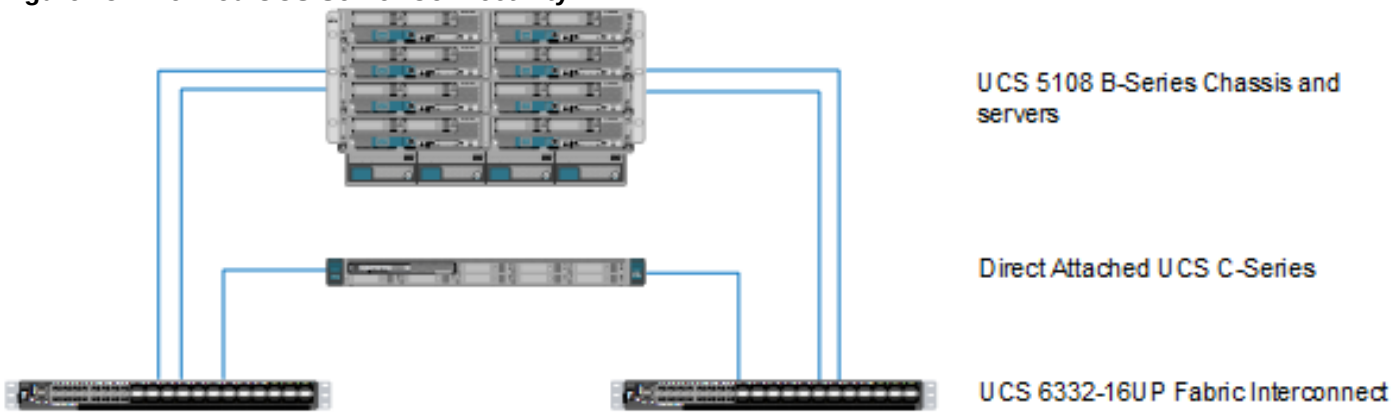
FlexPod UCS Design

The FlexPod design simultaneously supports both Cisco UCS B-Series and C-Series deployments. This section of the document discusses only the integration and design of Cisco UCS B-Series deployments into FlexPod.

FlexPod for ICP solution has been validated leveraging Cisco UCS M5 servers. Cisco UCS servers provide converged and highly available hardware platform centrally managed by Cisco UCS Manager software residing on Cisco Fabric Interconnects.

Cisco UCS servers are deployed with a single VIC card for unified network and storage access. The Cisco VIC connects into a redundant unified fabric provided by a pair of Cisco UCS Fabric Interconnects. Fabric Interconnects are an integral part of the Cisco Unified Computing System, providing unified management and connectivity to all attached chassis and blade servers. On a blade server chassis, Cisco UCS B-series servers connect into the fabric interconnect through Fabric Extenders (FEX) or Input Output Modules (IOM). Fabric Extenders extend the unified fabric from the FI to the chassis and serves as a consolidation point for all blade server I/O traffic. FEX is managed as an extension of the fabric interconnects, simplifying diagnostics, cabling and operations with a single point of management and policy enforcement.

Figure 23 FlexPod UCS Server Connectivity



FlexPod Network Design

This FlexPod design deploys a single pair of Nexus 9000 top-of-rack switches within each placement, using the traditional standalone mode running NX-OS.

Cisco Nexus 9000 provides Ethernet switching fabric for communications between the Cisco UCS domain, the NetApp storage system and the enterprise network. In the FlexPod design, Cisco UCS Fabric Interconnects and NetApp storage systems are connected to the Cisco Nexus 9000 switches using virtual Port Channels (vPC).

For validation, Cisco UCS B200 M5 blade servers with VIC 1340 adapters, were connected to 2 x Cisco UCS 6332 Fabric Interconnects. The Cisco UCS 5108 blade server chassis, housing the blade servers, were deployed using 2 x Cisco UCS 2304 FEX adapters to connect to the fabric interconnects. Two 40GbE links were used for FEX to FI connectivity, one from FEX-A to FI-A and one from FEX-B to FI-B, for an aggregate access bandwidth of 80Gbps from the blade server chassis to the unified fabric.

Connectivity from each individual fabric interconnect, to all upstream or northbound (NB) networks is provided by 2 x 40G links to each of the top-of-rack Cisco Nexus switches as follows:

- 2 x 40G uplinks from FI-A to Nexus-A and Nexus-B respectively
- 2 x 40G uplinks from FI-B to Nexus-A and Nexus-B respectively

Both uplinks are configured into a single port channel, making the total aggregate bandwidth to the core switching infrastructure 80Gbps per fabric interconnect. Each port designated as a core switch connection is designated as an uplink port within Cisco UCS Manager.

The switches are configured as vPC peers. vPCs are used to provide switch-level redundancy to the Cisco UCS fabric interconnects and AFF systems without requiring special configuration on those devices. The switches in this solution are operating in NX-OS mode but could also be configured as leaves in an ACI network.

Figure 24 FlexPod Network Design

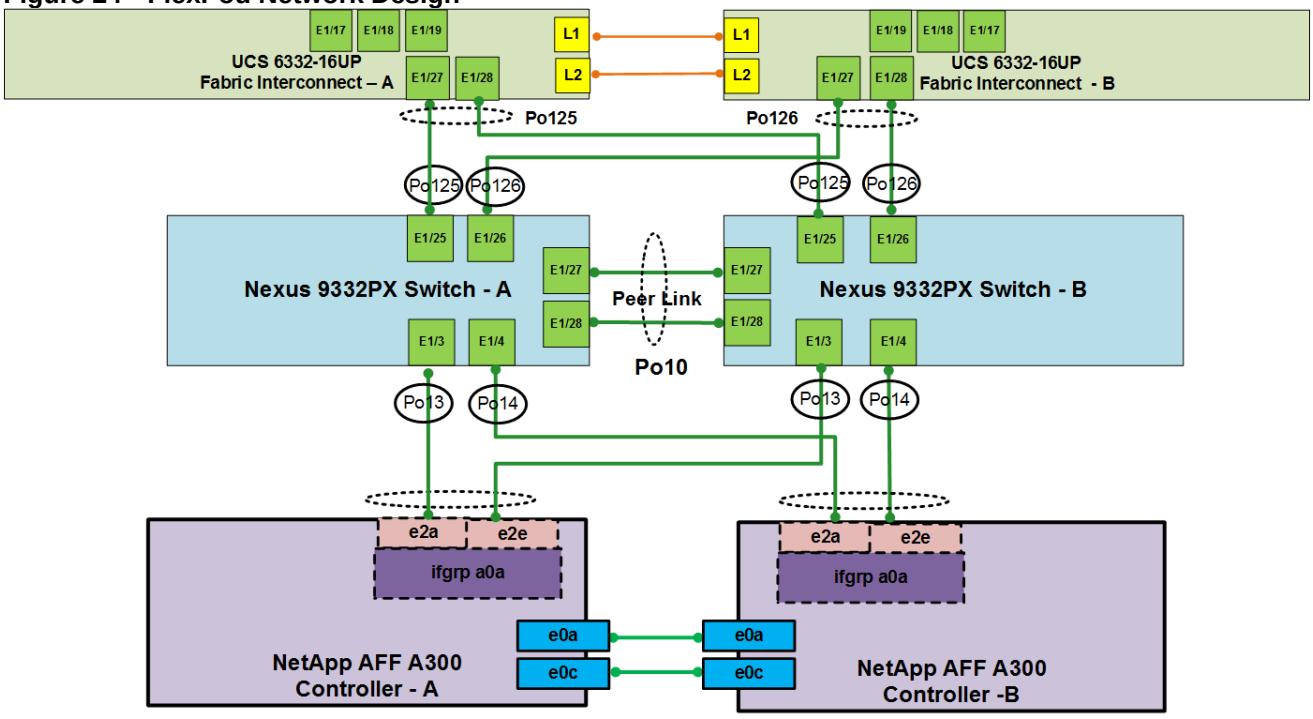
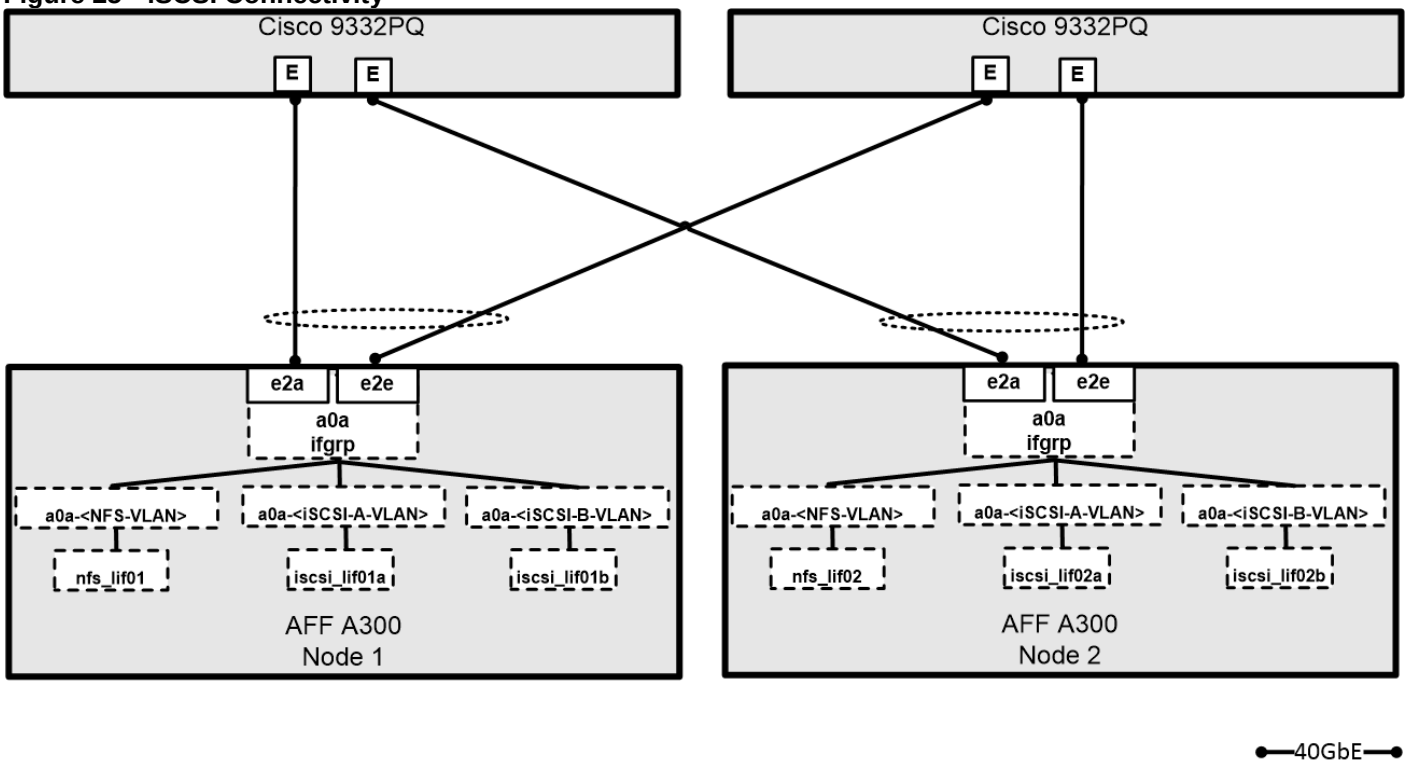


Figure 24 shows the connections between Cisco Nexus 9000, Cisco UCS Fabric Interconnects and NetApp AFF A300. vPC requires a “peer link” which is documented as port channel 10 in this diagram. In addition to the vPC peer-link, the vPC peer keepalive link is a required component of a vPC configuration. The peer keepalive link allows each vPC enabled switch to monitor the health of its peer. This link accelerates convergence and reduces the occurrence of split-brain scenarios. In this validated solution, the vPC peer keepalive link uses the out-of-band management network.

NetApp AFF A300 – iSCSI Connectivity

The storage controller’s 40GbE ports are directly connected to Cisco Nexus 9332PQ switches. Each controller is equipped with 40GbE cards on the expansion slot 2 that has two physical ports. Each storage controller is connected to two SAN fabrics. This method provides increased redundancy to make sure that the paths from the host to its storage LUNs are always available. Figure 25 shows the port and interface assignments connection diagram for the AFF storage to the Cisco Nexus 9332PQ SAN fabrics. This FlexPod design uses the following port and interface assignments. In this design, NFS and iSCSI traffic utilize the 40G bandwidth.

Figure 25 iSCSI Connectivity



Cisco UCS Server – Virtual Networking Design

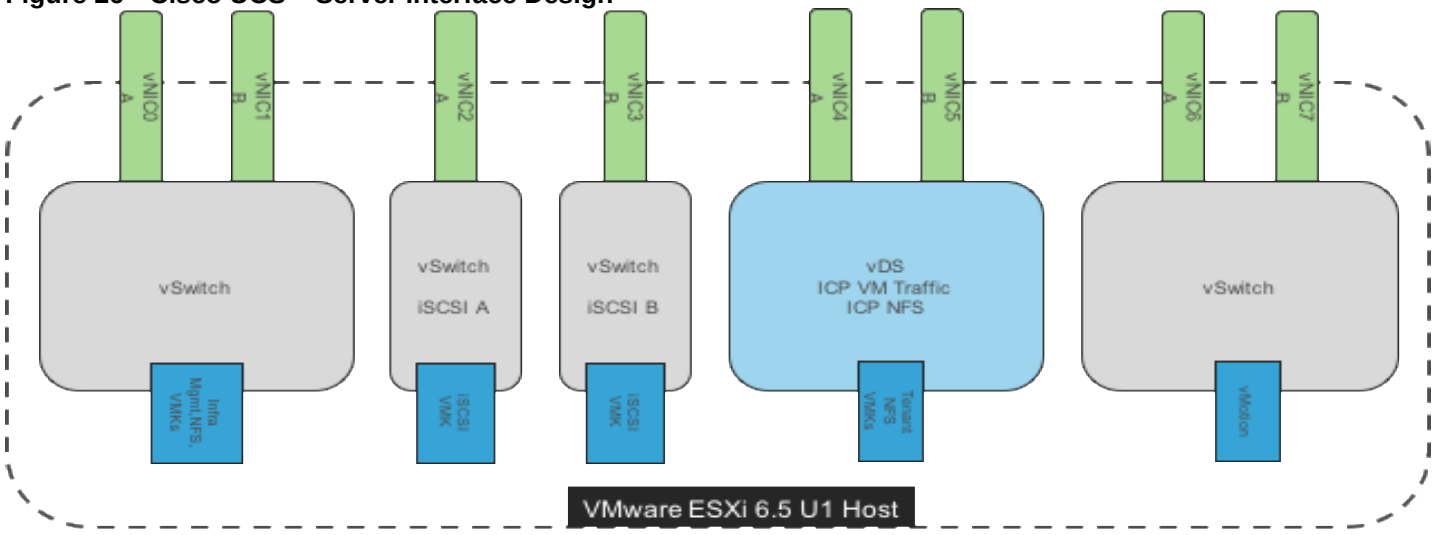
The ESXi nodes consist of Cisco UCS B200-M5 series blades with Cisco 1340 VIC. These nodes are allocated to a VMware High Availability (HA) cluster to support IBM cloud private infrastructure, services and applications. At the server level, the Cisco 1340 VIC presents multiple virtual PCIe devices to the ESXi node and the vSphere environment identifies these interfaces as vmnics or vmhbas. The ESXi operating system is unaware of the fact that the NICs or HBAs are virtual adapters.

In this design, the following virtual adapters are used with A vNICs connected to unified fabric A and B vNICs to unified fabric B resulting in each ESXi node being dual homed to the external network. The virtual adapters are assigned to different virtual switches depending on the type of traffic. VMware vSwitch is used for management, vCenter HA and vMotion traffic. VMware vDS is used for application traffic with different application port groups to isolate the applications. The iSCSI interfaces used for SAN boot of the ESXi server will need to be presented as native VLANs on dedicated vNICs that are connected with standard vSwitches

- One vNIC for iSCSI-A traffic
- One vNIC for iSCSI-B traffic
- Two vNICs for in-band management traffic
- Two vNICs for vMotion traffic
- Two vNICs for application related data including storage access if required. These vNICs are assigned to a distributed switch (vDS)

The ESXi management VMkernel port, the Infrastructure and NFS VMkernel ports can be transitioned to the VMware vDS, though the architecture for this solution validation has these ports left on standard vSwitches.

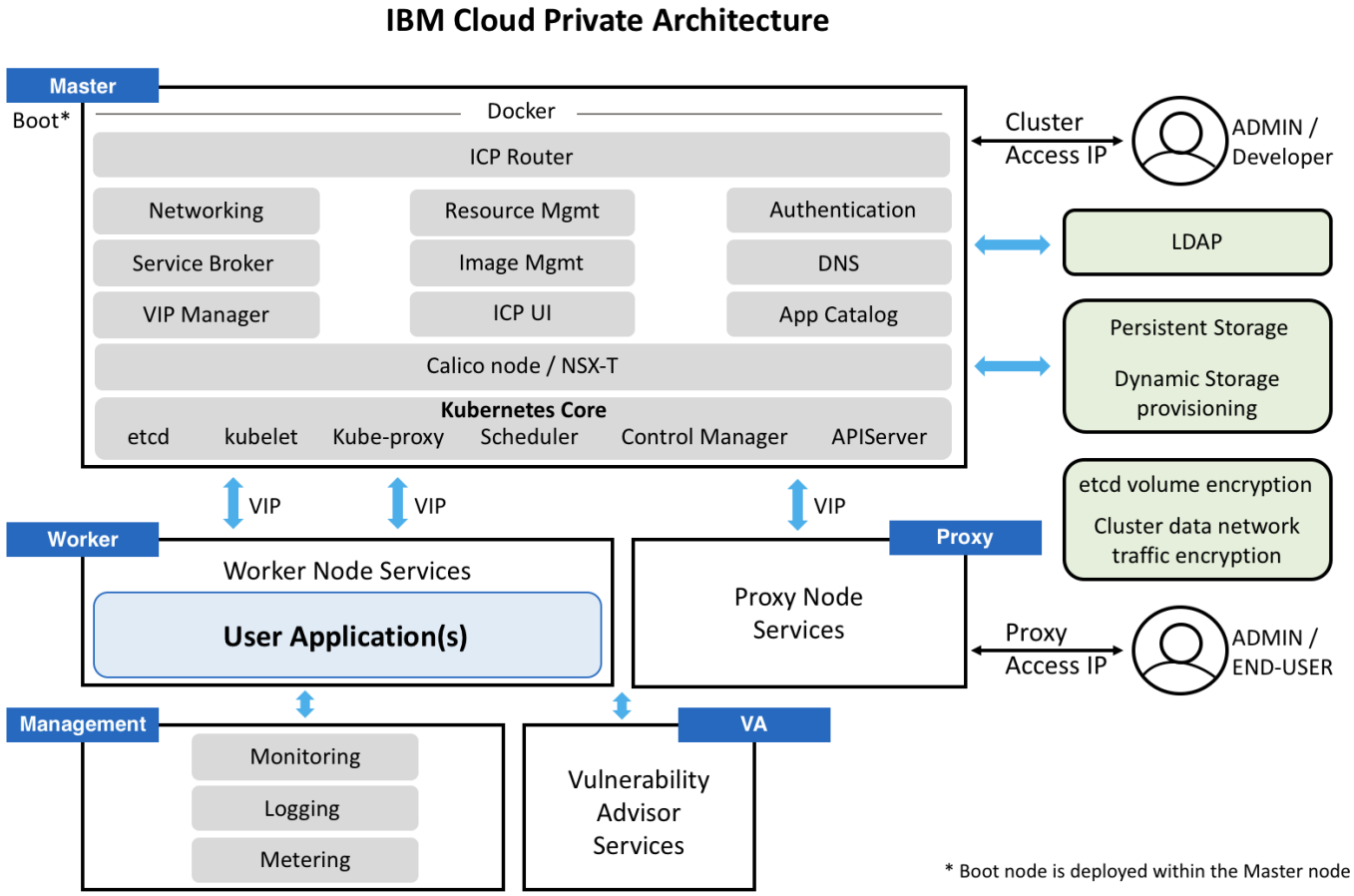
Figure 26 Cisco UCS – Server Interface Design



IBM Cloud Private Architecture

For the purpose of the CVD, the team deployed a highly available and scalable medium size ICP Kubernetes cluster. The nodes are deployed as VMs on supported guest OS, Ubuntu 16.04 in our case. Figure 27 illustrates the architectural view of ICP as deployed in our test environment.

Figure 27 IBM Cloud Private - Architecture



ICP is deployed on the hypervisor as a set of VMs. Each VM is a node with a role to carry specific functions within the ICP-Kubernetes cluster.

Boot Node
A boot or bootstrap node is used for running installation, configuration, node scaling, and cluster updates. Only one boot node is required for any cluster. A single node can be used as both master and boot, which is the case in our test environment with Boot node functionality included in the Master1 node.

Master Node
A master node provides management services and controls the worker nodes in a cluster. Master nodes host processes that are responsible for resource allocation, state maintenance, scheduling, and monitoring. Multiple master nodes are required in a high availability (HA) environment to allow for failover if the leading master host fails. Host that can act as the master are called master candidates. There are three Master nodes in our test environment to provide HA cluster, each is deployed on a separate host (ESXi) for redundancy.

Proxy Node
A proxy node is a node that transmits external request to the services created inside your cluster. Multiple proxy nodes are deployed in a high availability (HA) environment to allow for failover if the leading proxy host fails. A single node can be used as both master and proxy. However, it is best to have dedicated proxy nodes. A cluster must contain at least one proxy node if load balancing is required inside the cluster. Three Proxy nodes are deployed in our test environment to provide redundancy and service capacity for a medium size implementation.

Management Node
A management node is an optional node that only hosts management services like monitoring, metering, and logging. Having dedicated management node prevents the master node from becoming overloaded. Management node can be enabled only during the installation of ICP. In our test environment there are two management nodes for HA purposes.

Worker Node
A worker node is a node that provides a containerized environment for running tasks. As demands increase, more worker nodes can easily be added to the cluster to improve performance and efficiency. A cluster can contain any number of worker nodes, but a minimum of one worker node is required. There are five worker nodes in our initial test environment as recommended by IBM for minimal configuration of a medium size deployment. A sixth Worker node was added during the testing procedures to verify the scalability of the solution.

Additional VMs can be deployed to host enterprise applications that may be included in the Enterprise version of ICP (with the enterprise bundle software). IBM's Cloud Automation Manager can be used to deploy these VMs if not deployed via Helm charts.

IBM Cloud Private Deployment on FlexPod

For the validated design effort the team focused on a building and testing medium size ICP environment. Four UCS servers for four ESXi servers were installed and configured within vCenter to host the ICP nodes, all were Ubuntu 16.04 servers as the guest OS VMs.

Table 3 lists the number of nodes and the associated specifications that were deployed in this solution.

Table 3 ICP Nodes and Specifications

Node Type	Number of Nodes	CPU	Memory (GB)	Disk (GB)*
Boot	Included with Master Node			
Master	3	8	32	300
Management	2	4	16	300
Proxy	3	4	8	300
VA	1	4	16	300
Worker	5**	4	16	300

* Storage was thin-provisioned

** a 6th Worker node was added to validate scalability

The number of nodes and roles adhere to IBM's architecture as described in the following reference guide: https://github.com/ibm-cloud-architecture/refarch-privatecloud/blob/master/Sizing.md

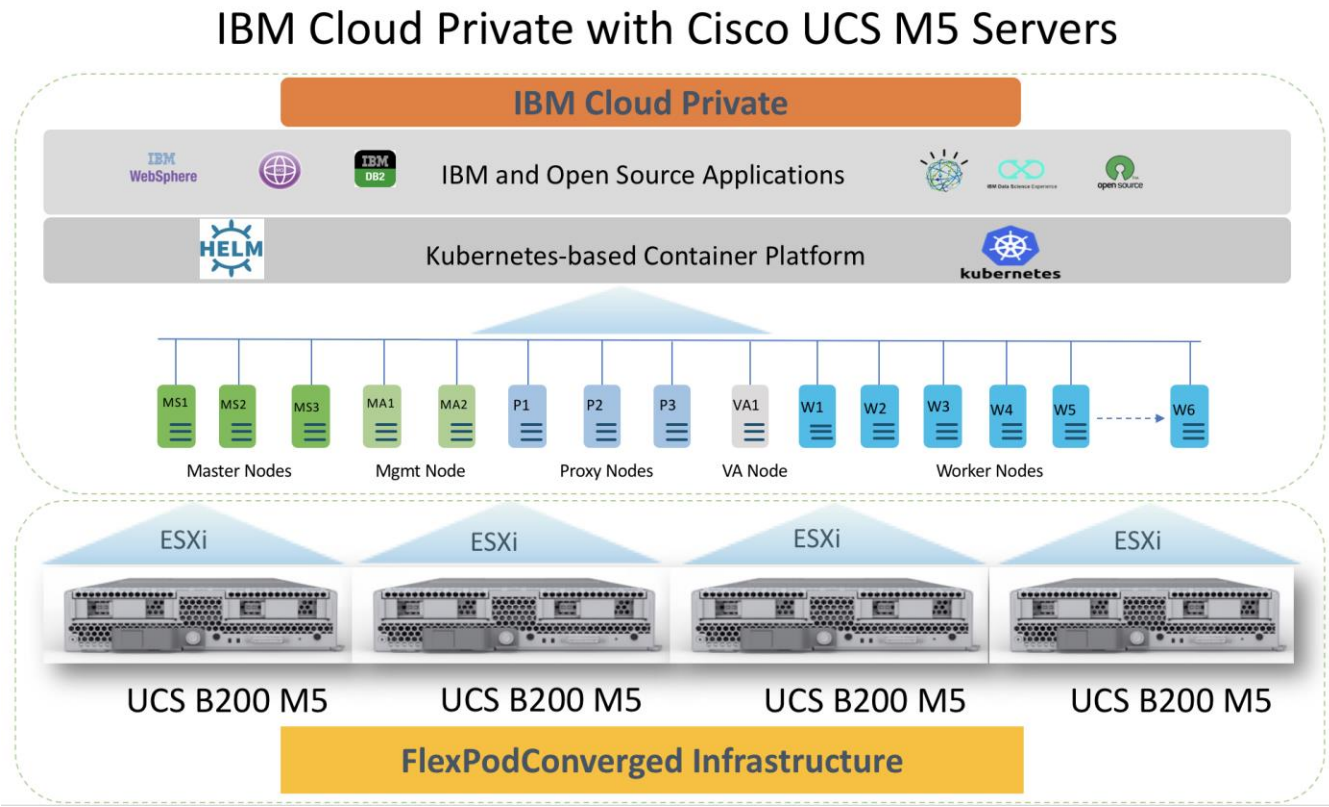
There are two exceptions:

1. Boot node role is included with the Master1 node

2. Only one node was configured for the Vulnerability Advisor (VA), which means that the VA node did not have HA configuration in our test environment. IBM does recommend 3 VA nodes for medium size cluster of ICP.

All nodes were configured as an Ubuntu 16.04 VMs, running on the ESXi servers as depicted in Figure 28.

Figure 28 ICP Nodes



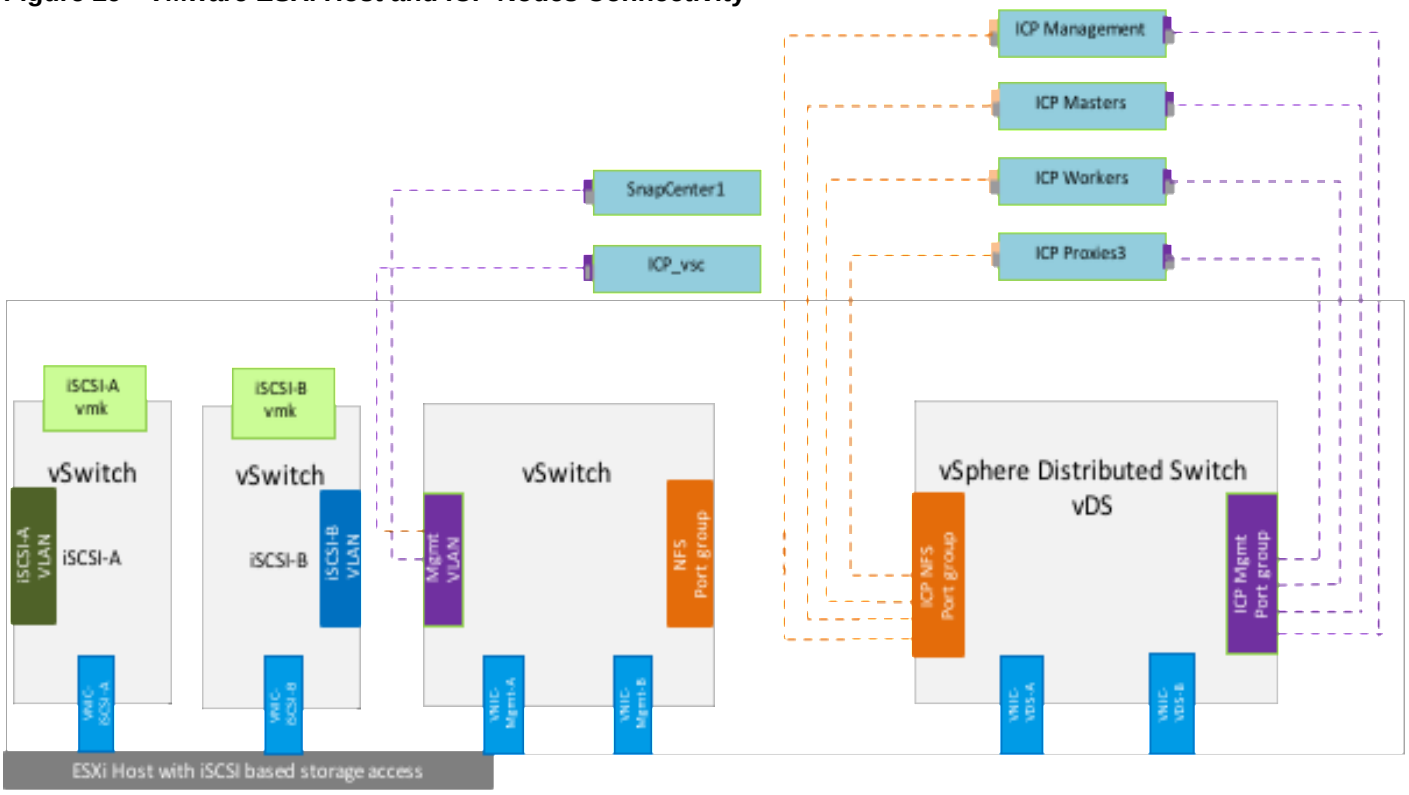
FlexPod VMware vSphere Design for ICP Communication

The compute resources in this architecture are grouped into a VMware ESXi infrastructure cluster. Servers in the cluster host the virtual machines used for ICP infrastructure deployment. High availability features available in VMware vSphere are leveraged to provide virtualization layer resiliency.

The VMware ESXi servers have, In-band management and vMotion traffic handled by infrastructure services vSwitches and iSCSI-A and iSCSI-B traffic is handled by two dedicated iSCSI vSwitches. The ESXi host configuration therefore has a combination of four vSwitches and a single distributed switch which handles application specific traffic.

IBM Cloud Private deployment utilizes one dedicated port group on the application VMware distributed switch (VDS) within the VMware ESXi servers on FlexPod. The ICP management and Worker nodes have two virtual NICs, one NIC connected to ICP port group for communication between the ICP nodes, the other NIC connected to the NFS port group to enable NFS storage access to the NetApp AFF A 300. The Master nodes share a common NFS share to store the configuration data for high-availability and the ICP worker nodes need NFS storage access for dynamic storage provisioning.

Figure 29 VMware ESXi Host and ICP Nodes Connectivity



Considerations

The following sections outline the design considerations for the FlexPod with IBM Cloud Private solution.

Resiliency

The FlexPod for ICP solution addresses infrastructure resiliency by including redundancy in its design and implementation at the level of each component (compute, network and storage). Design considerations and best practices associated with fault tolerance, resiliency and other redundancy aspects to help ensure high availability of the converged infrastructure are addressed in a previously published CVDs which can be found from the following links:

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/flexpod_esxi65u1design.html

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/flexpod_esxi65u1_n9kiscsi.html

Additional crucial element of resiliency is provided at the hypervisor level (VMware vSphere) managed by vCenter. In our specific reference there are four physical servers for the four ESXi hosts. Three are required in order to support three master nodes, which is the required number of nodes for highly available ICP platform. The fourth host is then use for high availability and is redundant until a failure. The fourth server also supports additional workload such as management tools and can be used also to host additional Worker nodes. It is important to size the entire set of compute, network and storage to accommodate the expected workloads not only from the perspective of the ICP nodes but also to account for user applications.

Mapping the ICP nodes to an ESXi server is not important and infrastructure teams can rely on the hypervisor layer to automatically manage the VMs affectively across the available pool of resources.

It is critical however to prevent a situation in which the Master nodes, that have critical role in the availability of the service, from being hosted on the same underlying ESXi server. Only one Master node should be allowed on each

of the ESXi servers. Since there are three Master nodes in our cluster, we will need a fourth ESXi server as a standby to host nodes from a failed ESXi. Given the number of total nodes and expected total workloads, our recommendation is to have the fourth ESXi server in order to have the proper redundancy. That makes the topology of physical ESXi servers as N + 1. In larger environments with more nodes the topology may call for N + 2 or more.

Infrastructure teams can configure rules in the VMware environment to manage this level of resiliency. In our CVD testing the team configured VMware DRS Anti-Affinity rules to make sure that no more than one Master node will be hosted per ESXi server.

The team also tested an ESXi failure and observed the expected behavior of the impacted running nodes failing over to other available resources in the pool, including the fourth ESXi hosting the Master server from the failed ESXi.

VMware vMotion provides the functionality of moving VMs automatically across the available pool of resources and according to set of rules if defined by the administrator. vMotion networks were also defined on the ESXi host to handle the associated network traffic.

Scalability

FlexPod is highly scalable and flexible converged infrastructure stack from which ICP customers can benefit. At the converged infrastructure FlexPod level the various components can scale easily, within the same element management. Cisco UCS servers can be easily added, additional network ports and modules can be added and also storage capacity and additional controllers can be added, all in a non-disruptive fashion, allowing organization to scale easily as they add data and services to the platform. Without impacting the availability of the service.

In some cases scalability will have multiple dimensions, so not just vertically by adding more memory or capacity to the storage system but also horizontally by adding more units of compute, network ports and storage controllers. New UCS units may include higher core count and more memory than previously deployed servers, which will support higher total of workload. This mix architecture is applicable also at the storage level as the ONTAP cluster can support different FAS/AFF models acting as one cluster and different media type can also be included (such as NL-SAS).

This flexibility in how the FlexPod platform can scale helps organization optimize the balance between performance and cost and allow them to allocate the right amount of resources to the required task and change that easily if and when needed.

Sizing and Performance

The solution covered in this reference architecture is based on a medium size ICP environment as defined by IBM in the following document: https://github.com/ibm-cloud-architecture/refarch-privatecloud/blob/master/Sizing.md and as listed in Table 4 . A medium size ICP environment accounts for highly available cluster for resiliency of the service and also for its ability to handle loads and scale, hence it suggests to separate nodes with different roles and run them as individual VMs.

Table 4 Medium Size ICP Environment

Node type	Number of nodes	CPU	Memory (GB)	Disk (GB)
Boot	1	2	8	250
Master	3	8	32	250
Management	2	8	32	300
Proxy	3	4	16	250
VA	3	6	24	500
Worker	5+ (Max:70)	8	32	250
Total	17+	112	448	5100

For our CVD testing, the team followed the recommendations from IBM regarding the number of nodes with the following deviations:

1. Only one VA node was deployed, since validating the high availability of the service was not covered in the scope of this CVD

2. Boot node was deployed as part of the Master1 node for simplicity

This is a general recommendation and not specific to a customer environment. As such, it is important to properly size the solution with all of its components by a qualified Sales Engineer / Architect per the specific requirements of the customer. There is no one size fits all approach, hence specific sizing and performance testing were excluded from the CVD. However, Cisco, NetApp, IBM, and their partners, do all provide tools and/or resources to help organizations optimize the sizing of the solution to meet the required performance in the most economical way.

Certain features of the solution can help optimize cost/performance. At the storage level ONTAP includes QoS feature, which will help organizations prioritize certain workloads and meet critical service levels. At the Kubernetes level customers can define different storage classes that translate to different storage services with different characteristics at the backend ONTAP platform. Although in our current reference architecture we are focusing on All Flash technology, in some situations customers may want to mix media and storage controllers that support also NL-SAS as an example, and structure the storage services that offer low latency and high performance as well as lower performance higher latency for some workloads.

At the Cisco UCS level, customers have the option to include servers with different processors and core counts, and with the combination of the right amount of memory the servers can be optimized for the right cost-performance configuration.

It is important to size the servers to meet the minimal requirements of the ICP platform, to account for failures of servers and by that to make sure that VMware DRS related rules can be followed upon server failure with enough resources available for VMware to redistribute the VMs from the failing host or when performing upgrades and other maintenance tasks.

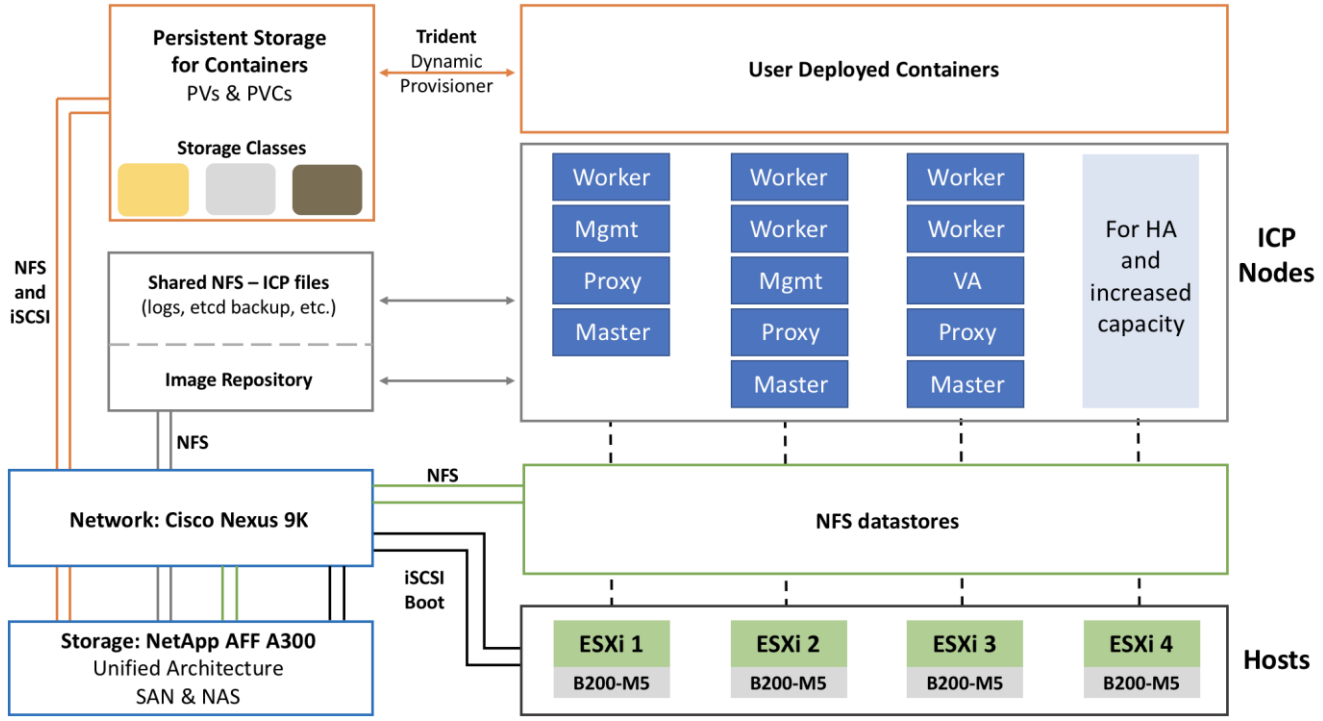
Additional information about performance on scaling deployments is provided in the Validation section of this document.

Solution Deployment

Architecture

This section provides details about the deployment of ICP on the FlexPod CI as tested and validated in our lab. It focuses on configurations and settings that are specific to ICP and it does not cover generic details on how to set up the hardware components of the FlexPod or vSphere, which were covered in previously published CVDs and other technical publications from Cisco and NetApp. Figure 30 provides a diagram of the logical topology and is brought here as background and an introduction, prior to the details covered in the sub-sections.

Figure 30 FlexPod for ICP – Logical Topology



Deployment Hardware and Software

The deployment of hardware and software for FlexPod with IBM Cloud Private is detailed in the following sections.

The existing deployment of the FlexPod architecture is assumed, and the setup of these resources will have dependencies covered in the FlexPod Datacenter with VMware vSphere 6.5, NetApp AFF A-Series, and IP-Based Storage deployment guide available here:

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/flexpod_esxi65u1design.html

Deployment Hardware

Table 5 describes the hardware and software versions used during solution validation. It is important to note that Cisco, NetApp, and VMware have interoperability matrixes that should be referenced to determine support for any specific implementation of FlexPod. Click the following links for more information:

- NetApp Interoperability Matrix Tool: http://support.netapp.com/matrix/

- Cisco UCS Hardware and Software Interoperability Tool: http://www.cisco.com/web/techdoc/ucs/interoperability/matrix/matrix.html
- VMware Compatibility Guide: http://www.vmware.com/resources/compatibility/search.php

Table 5 Hardware and Software Revisions Validated

Layer	Device	Image	Comments
Compute	Cisco UCS Fabric Interconnects 6300 Series, Cisco UCS B-200 M5,	3.2(3d)	Includes the Cisco UCS-IOM 2304, Cisco UCS Manager, and Cisco UCS VIC 1340
	Cisco nenic Driver	2.3.0.10	Ethernet driver for Cisco VIC
	Cisco fnic Driver	1.6.0.34	FCoE driver for Cisco VIC
Network	Cisco Nexus Switches	7.0(3)I4(8a)	NXOS
Storage	NetApp AFF A300	ONTAP 9.3	Software version
	NetApp DS 224C Disk Shelf		Software version
	NetApp Trident	18.04.0	Software version
Software	VMware vSphere ESXi	6.5 update 1	Software version
	VMware vCenter	6.5 update 1	Software version
	IBM Cloud Private Enterprise Edition	2.1.0.2	Software version
	ICP Master Node	Linux Ubuntu 16.04 VM	Software version
	ICP Proxy Node	Linux Ubuntu 16.04 VM	Software version
	ICP Worker Node	Linux Ubuntu 16.04 VM	Software version
	ICP Management Node	Linux Ubuntu 16.04 VM	Software version

Configuration Guidelines

This document provides the details to configure a fully redundant, highly available configuration for a FlexPod unit with IBM Cloud Private environment. FlexPod infrastructure deployment is beyond the scope of this document and the following information is provided for reference. General reference is made to which component is being configured with each step, either 01 or 02 or A and B. For example, node01 and node02 are used to identify the two NetApp storage controllers that are provisioned with this document, and Cisco Nexus A or Cisco Nexus B identifies the pair of Cisco Nexus switches that are configured. The Cisco UCS fabric interconnects are similarly configured.

The focus of this document is to provide details of steps required to provision multiple Cisco UCS hosts and IBM Cloud Private nodes, and these examples are identified as: ESXi-Infra-Host-01, ESXi-Infra-Host-02 and ICP-Master-1, ICP-Master-2 etc to represent infrastructure hosts and ICP nodes deployed respectively in this document. Finally, to indicate that you should include information pertinent to your environment in a given step, <text> appears as part of the command structure. See the following example for the network port vlan create command:

Usage:

network port vlan create?

[-node] <nodename> Node

{ [-vlan-name] {<netport>|<ifgrp>} VLAN Name

| -port {<netport>|<ifgrp>} Associated Network Port

[-vlan-id] <integer> } Network Switch VLAN Identifier

Example:

network port vlan -node <node01> -vlan-name i0a-<vlan id>

This document is intended to enable you to fully configure the customer environment. In this process, various steps require you to insert customer-specific naming conventions, IP addresses, and VLAN schemes, as well as to record appropriate MAC addresses. Table 6 lists the VLANs necessary for FlexPod deployment and is provided for reference.

Table 6 Necessary VLANs

VLAN Name	VLAN Purpose	ID Used in Validating This Document
Out of Band Mgmt	VLAN for out-of-band management interfaces	13
In-Band Mgmt	VLAN for out-of-band management interfaces	113
iSCSI-A	VLAN for iSCSI-A Traffic	3010
iSCSI-B	VLAN for iSCSI-B Traffic	3020
Native	VLAN to which untagged frames are assigned	2
ICP Mgmt	VLAN for Container Management NFS traffic	901
ICP Tenant NFS	VLAN for First Container Tenant NFS traffic	3052

FlexPod Storage Configuration

The following storage elements were configured according to the Solution Design section of this document (0).

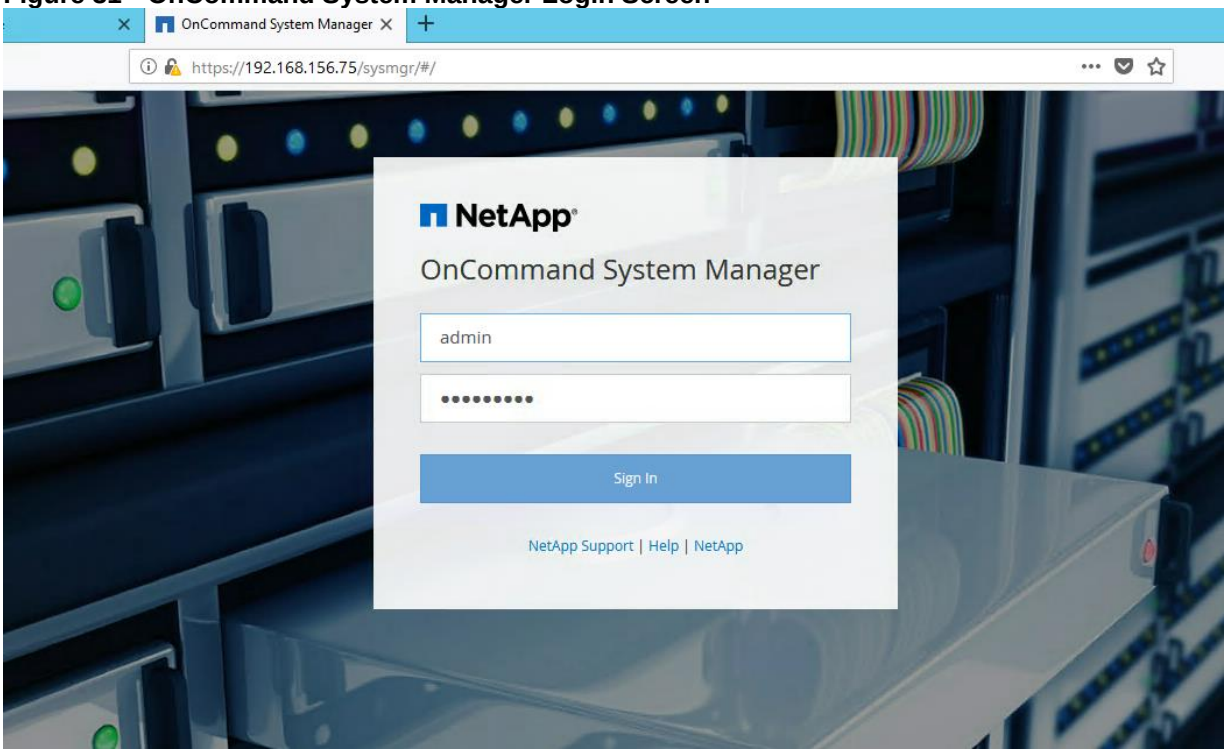
The initial setup and basic configurations of the storage system to meet the design are beyond the scope of this document and were covered in previously published CVD and other documents from NetApp. This section does provide information about the specific storage elements that were deployed and configured to benefit this solution.

A300 Storage System – Management Access

To configure and manage the various storage features and services we used the OnCommand System Manager UI. CLI is an alternative and can cover all configuration and settings options.

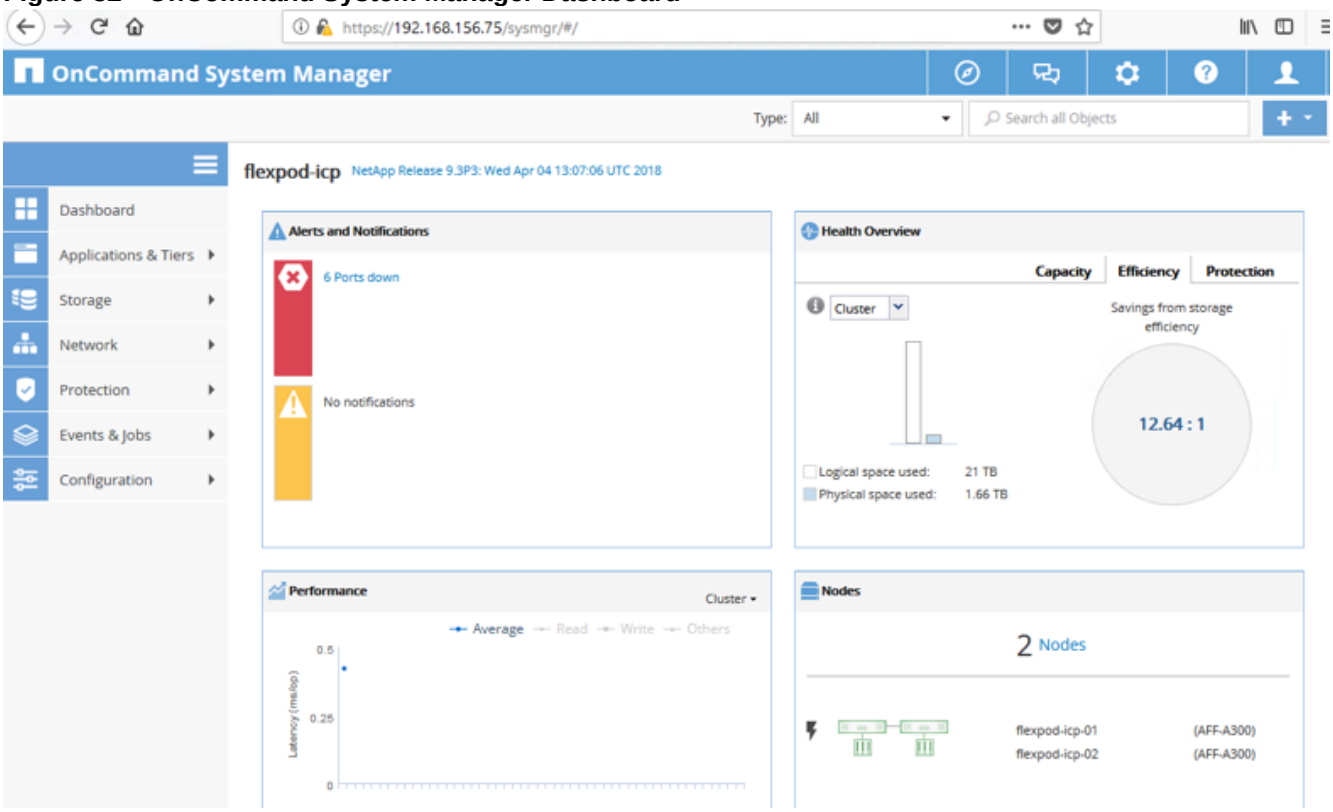
OnCommand System Manager, or System Manager, is accessible from a browser that points to the cluster management IP that was configured during the initial setup over https (https://192.168.156.75 in our system).

Figure 31 OnCommand System Manager Login Screen



When logged in, the dashboard screen is presented with the menu of options on the left side of the screen.

Figure 32 OnCommand System Manager Dashboard



Storage Controllers in HA Pair Configuration

As with all FlexPod CVDs, this solution also adheres to enterprise-grade requirements for redundancy and high availability (HA). As describe in the Technology Overview section, the AFF A300 storage system has a HA pair configuration which comprised of two storage controllers (2-nodes), and in the case of the A300 model, they are embedded within a single chassis. The nodes are both active, and in case of a failure of one, the other controller (partner) will take over for continuous operation and availability of all storage services.

The nodes were configured as `flexpod-icp-01` and `flexpod-icp-02` as listed in Figure 33of the Nodes view in the System Manager management UI.

Figure 33 HA Pair A300

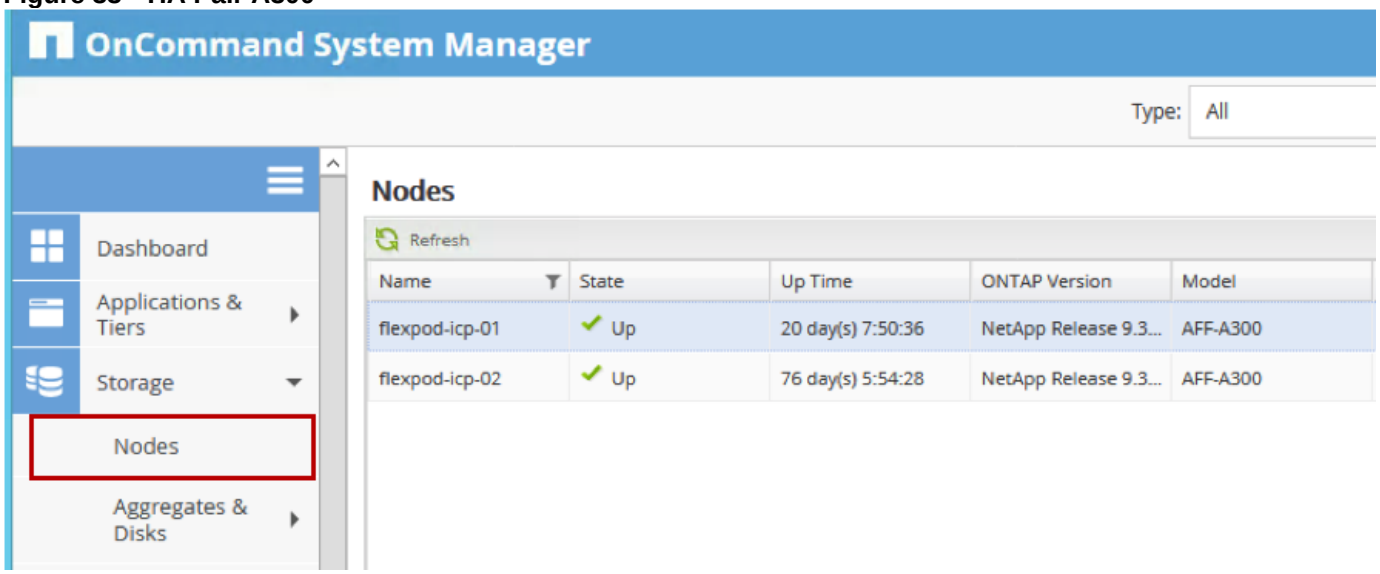
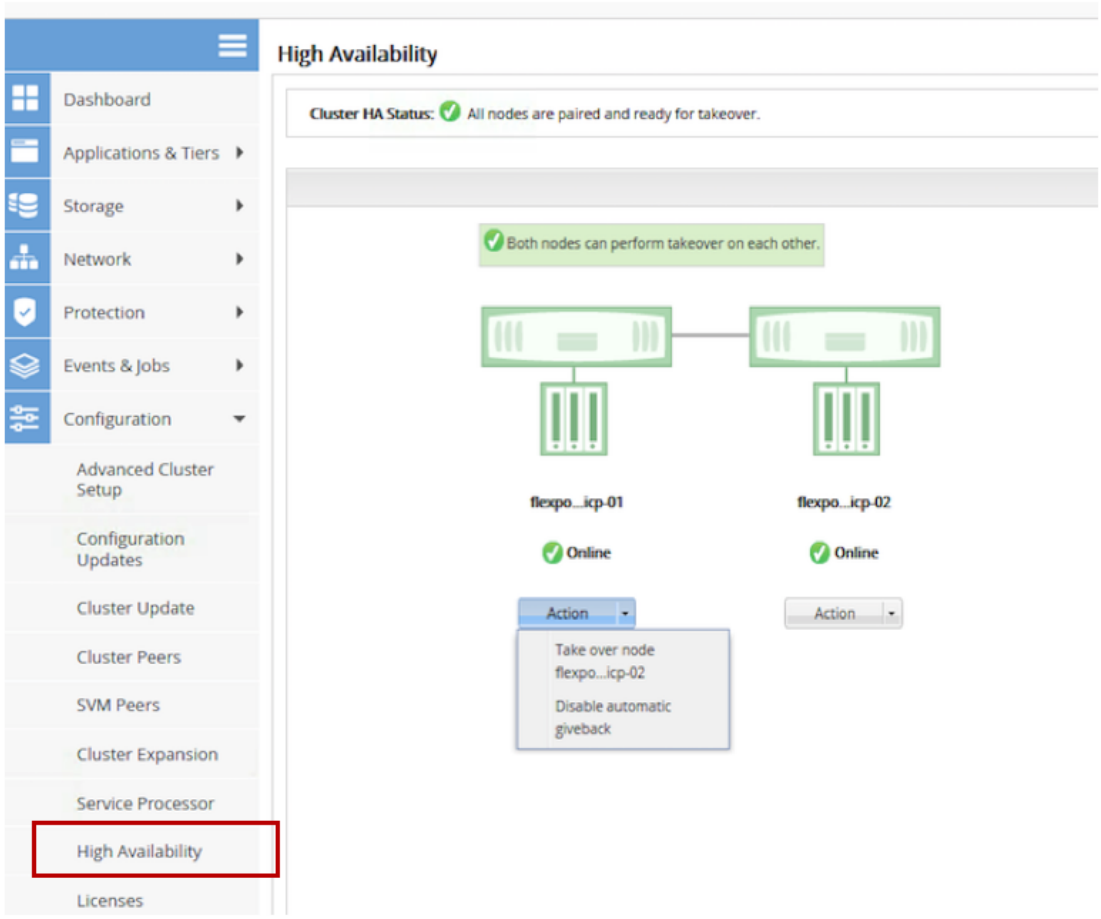


Figure 34 shows the high availability state of the cluster with both nodes online, ready to take over one another in case of a failure.

Figure 34 HA Status



Network Configuration of the A300

Segmentation of network traffic is best practice across all components of the infrastructure, including the storage system. The network settings of the A300 deployed in our test environments were as listed in Figure 35, indicating the Ethernet Ports, and in Figure 36 listing the Network Interfaces used in our test environment.

It is important to notice the following settings:

1. Cluster traffic is occurring over the cluster LIFs (Cluster Interconnect) over ports e0a and e0b on both nodes. These are the four interfaces: `flexpod-icp-01_clus1`, `flexpod-icp-01_clus2`, `flexpod-icp-02_clus1` and `flexpod-icp-02_clus2`

2. Four iSCSI interfaces for the SAN boot using ports `a0a-3010` and `a0a-3020`. The interfaces are: `iscsi_lif01a`, `iscsi_lif01b`, `iscsi_lif02a` and `iscsi_lif02b`. Also, please notice the SVM association.

3. Two NFS interfaces, one per node, were configured using ports `a0a-3050` on both nodes. The interfaces are: `nfs_lifs01` and `nfs_lifs02`

4. Additional management interfaces were created to simplify operation and traffic.

Figure 35 A300 Ethernet Ports

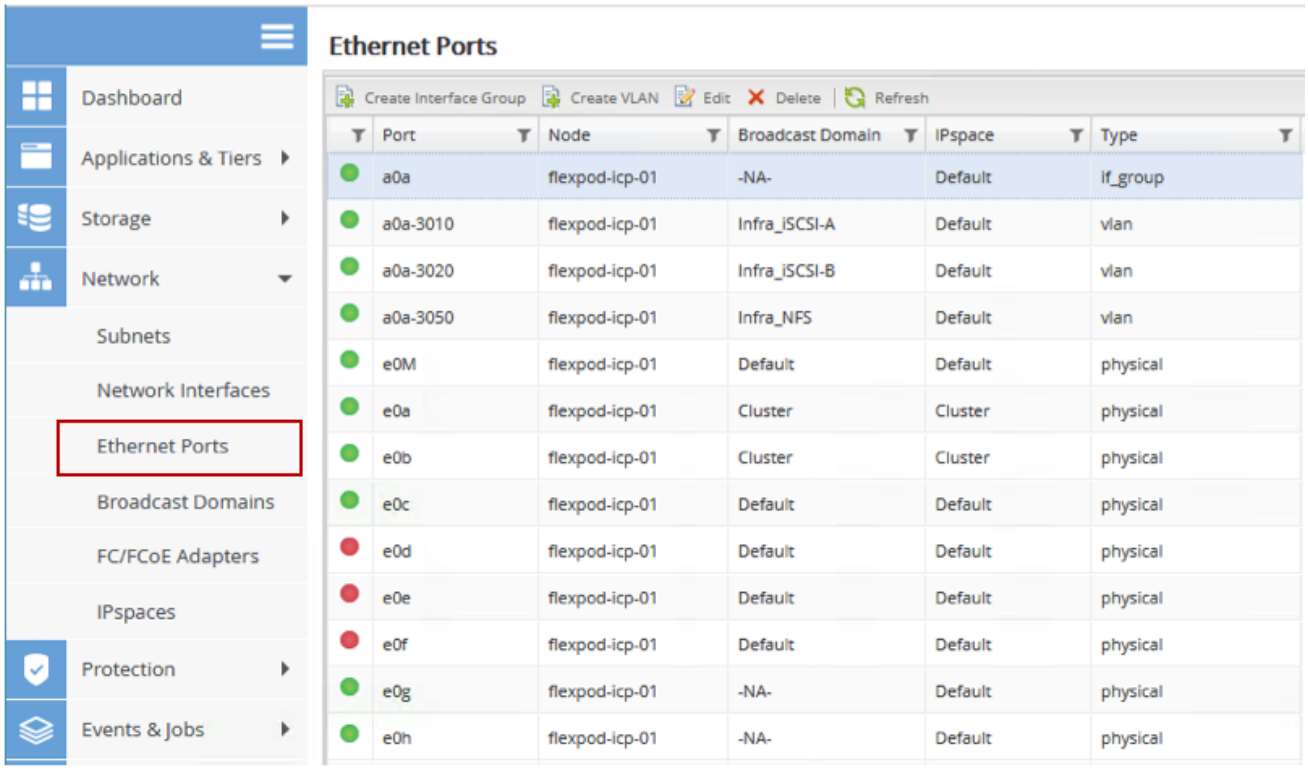
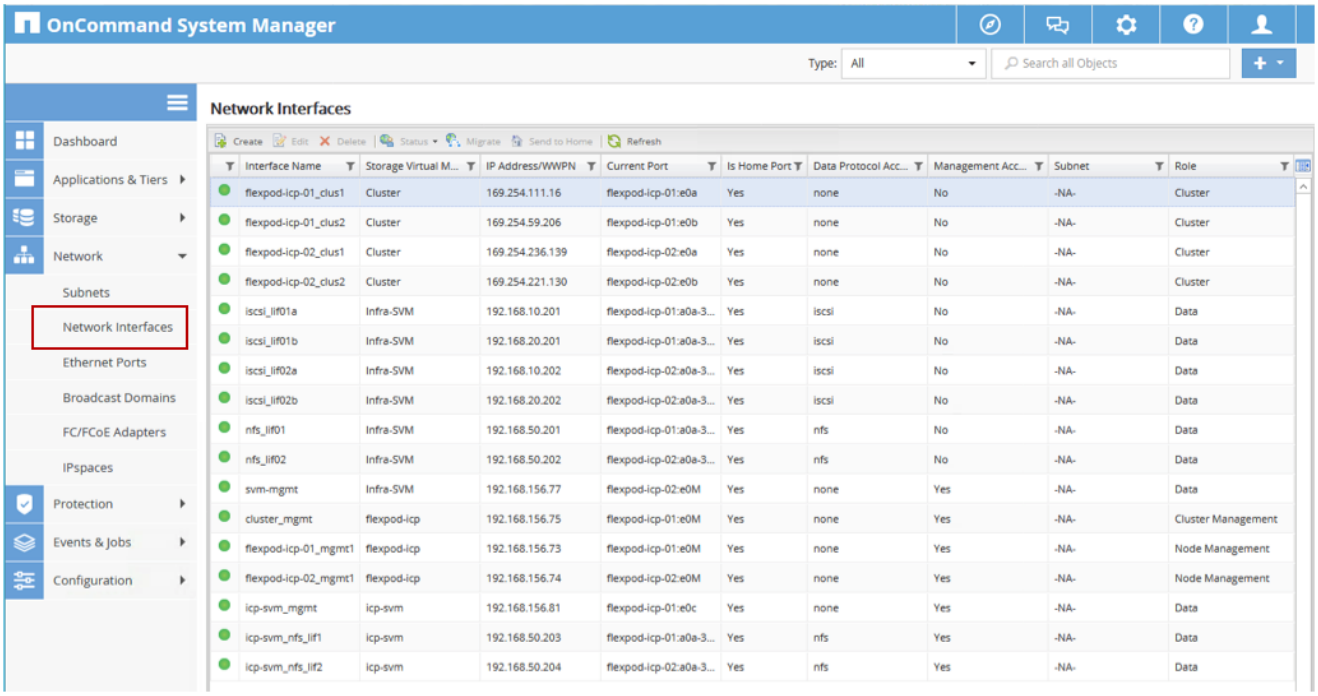


Figure 35 lists the ports of node 1. Node 2 has the exact port layout and configuration.

Figure 36 A300 – List of Network Interfaces

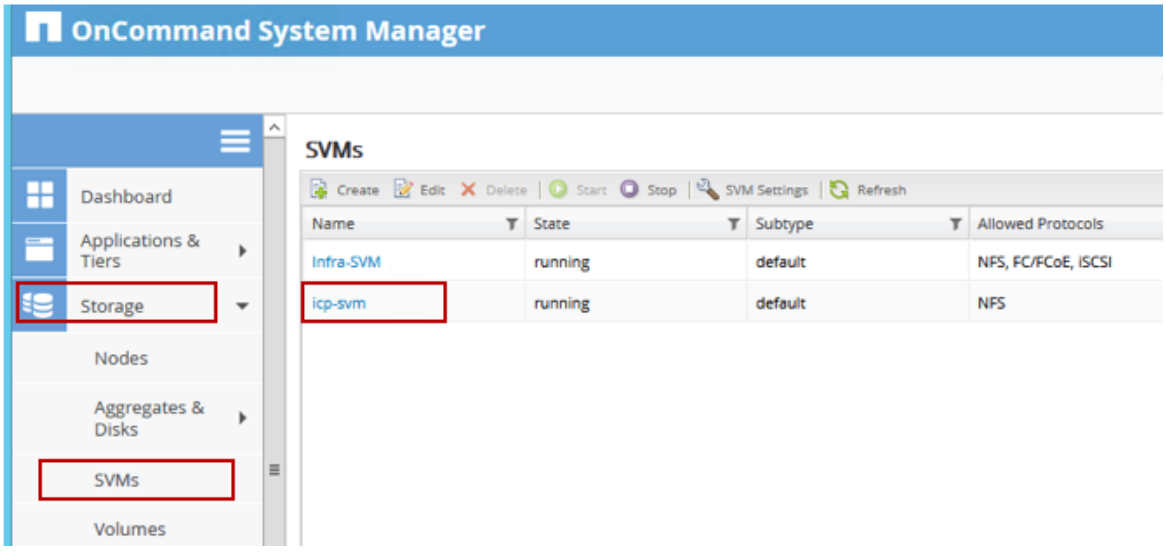


Storage Virtual Machines (SVM) Configuration

The storage system was configured with two SVMs: `Infra-SVM` for the iSCSI boot and the NFS datastores that contains the ICP nodes, and `icp-svm` for persistent volumes that will be provisioned via Trident. Implementation of SVMs will vary between one organization to another, please refer to the Technology Overview for more information about SVMs.

SVMs can be configured from OnCommand System Manager. Figure 37 lists the two SVMs created in our environment.

Figure 37 SVMs



We can examine the various settings of SVM `icp-svm`. Figure 38 lists the `vsadmin` user created for this SVM. It provides access to the specific SVM in case we want to implement certain access control. Different users with different roles can be added as needed.

Figure 38 SVM Users

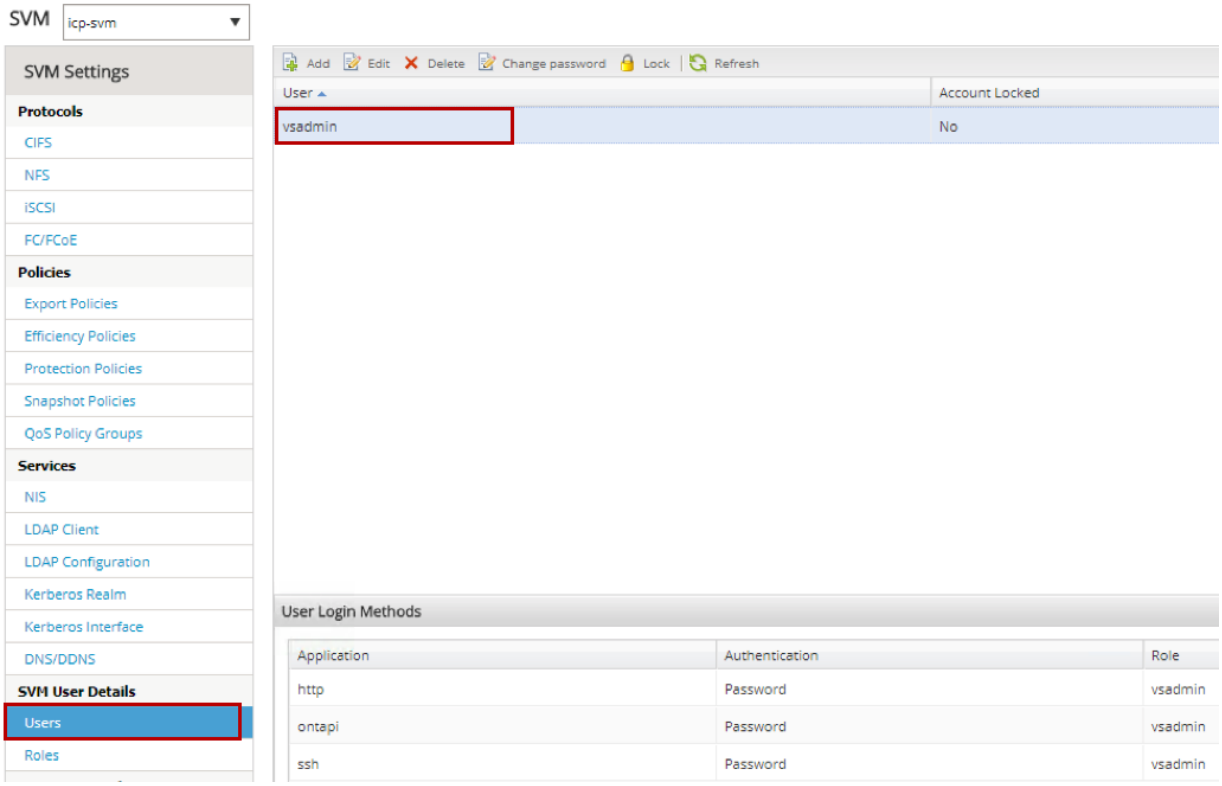


Figure 39 shows the export policies of `icp-svm`. Export policies were defined to provide specific access from the nodes to the shared NFS volumes.

Figure 39 NFS Export Policies

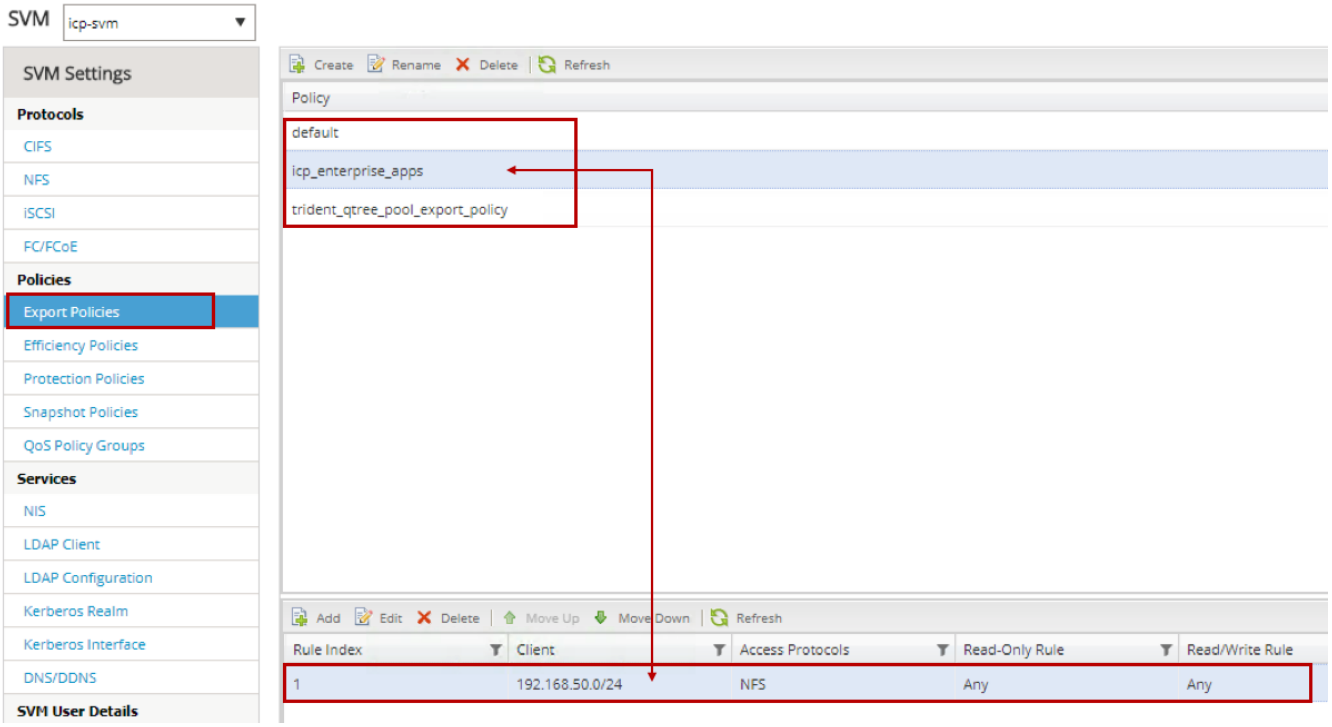


Figure 40 shows the storage efficiency policies associated with the SVM.

Figure 40 Storage Efficiency Policies

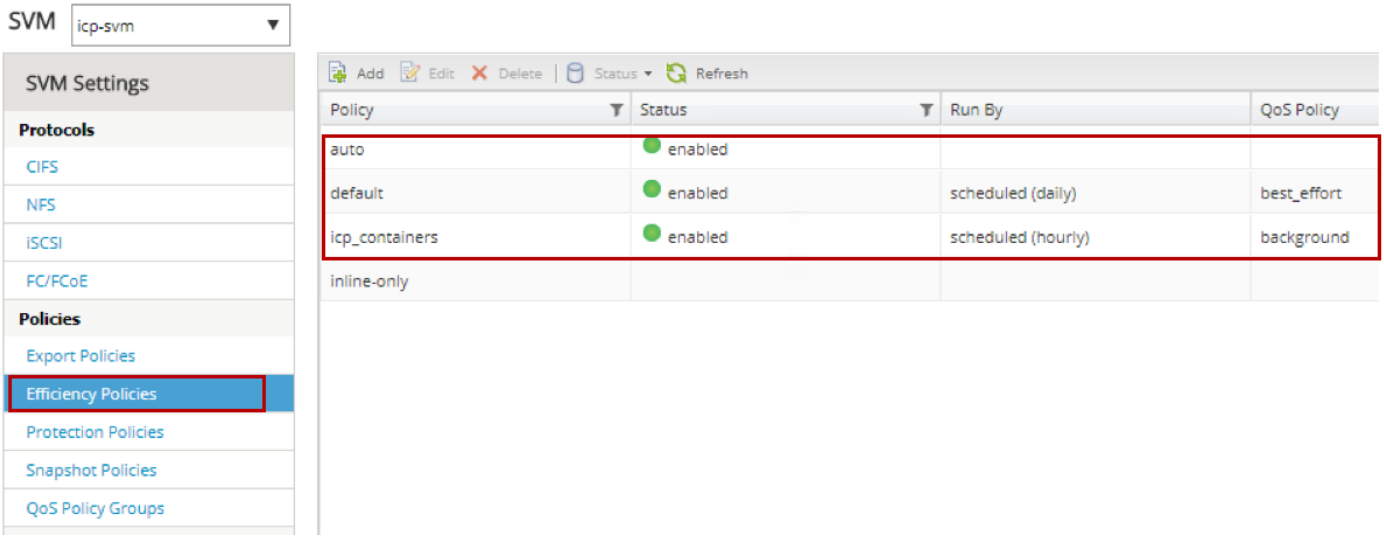


Figure 41 shows the defined Data Protection policies.

Figure 41 Data Protection Policies

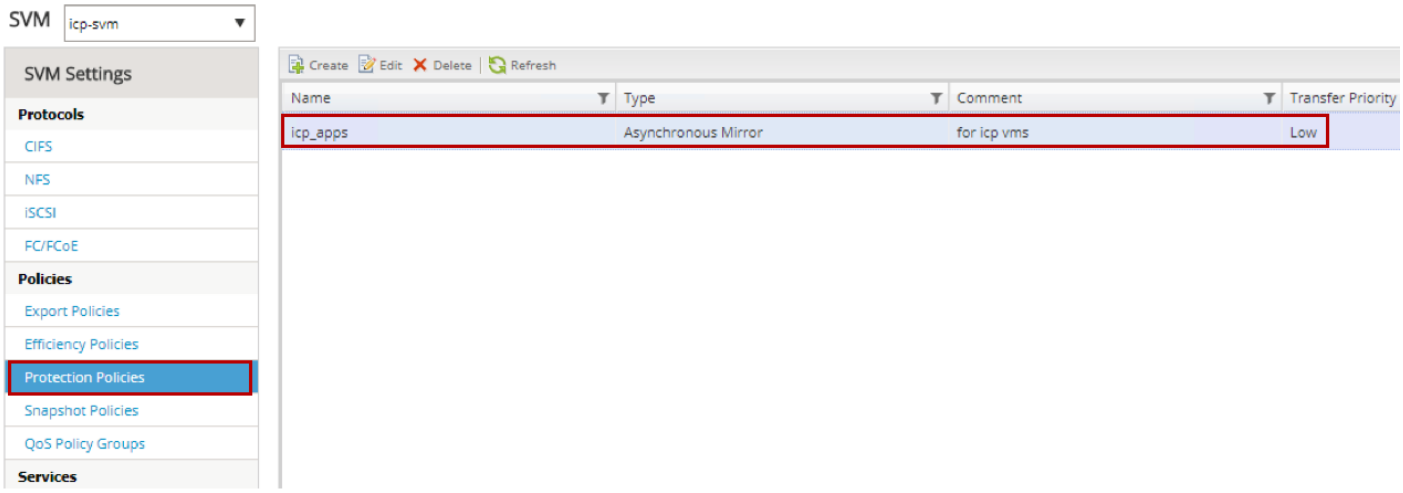
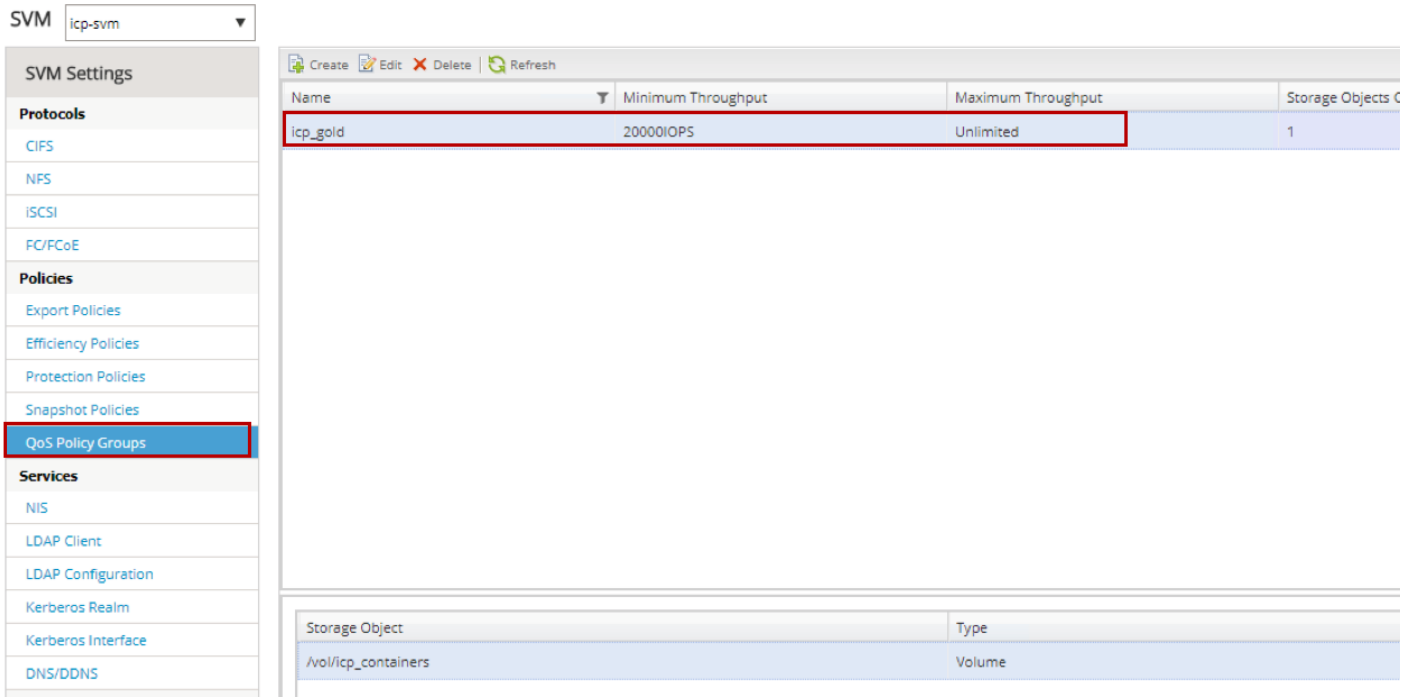


Figure 42 lists the QoS policy groups configured for this SVM. In this case, `icp_gold` listed as the name of the policy with minimum throughput set as 20,000 IOPS.

Figure 42 QoS Policy Groups



iSCSI LUNs

iSCSI LUNs were configured to support and enable SAN boot of the four ESXi servers. The LUNs are listed in Figure 43 and Figure 44 with the associated initiator groups. Notice the LUNs are associated with the infrastructure SVM, `Infra-SVM`, and not the `icp-svm` which was only configured with NFS.

Figure 43 iSCSI LUNs for ESXi

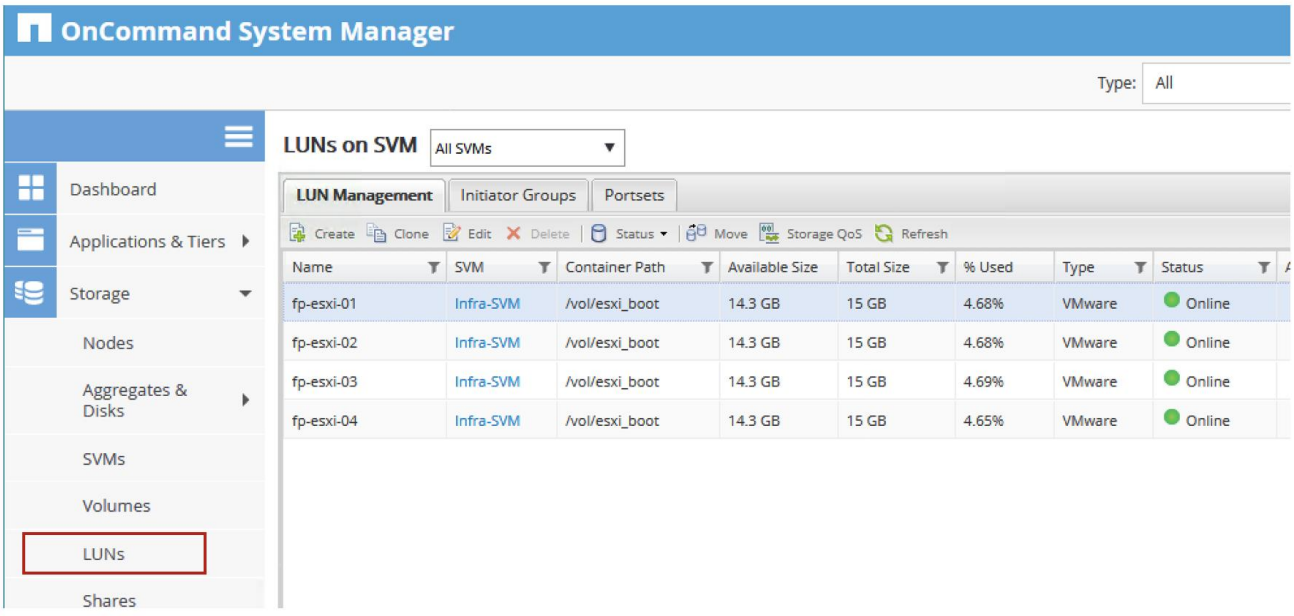
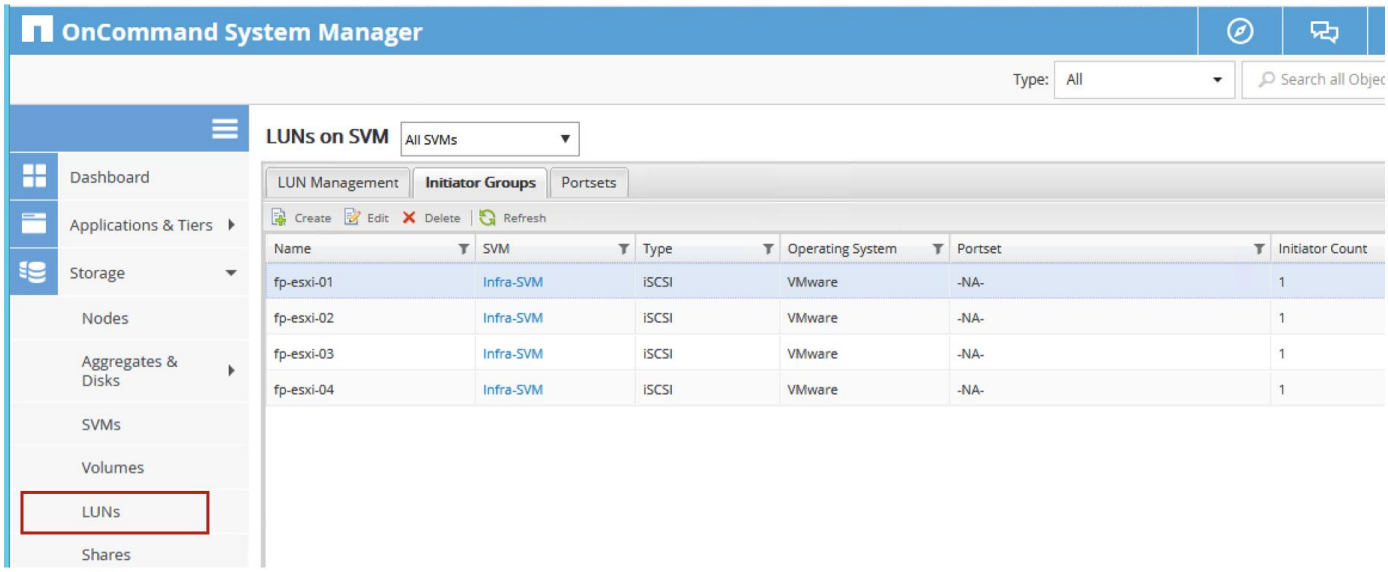


Figure 44 iSCSI Initiator Groups



Data Protection

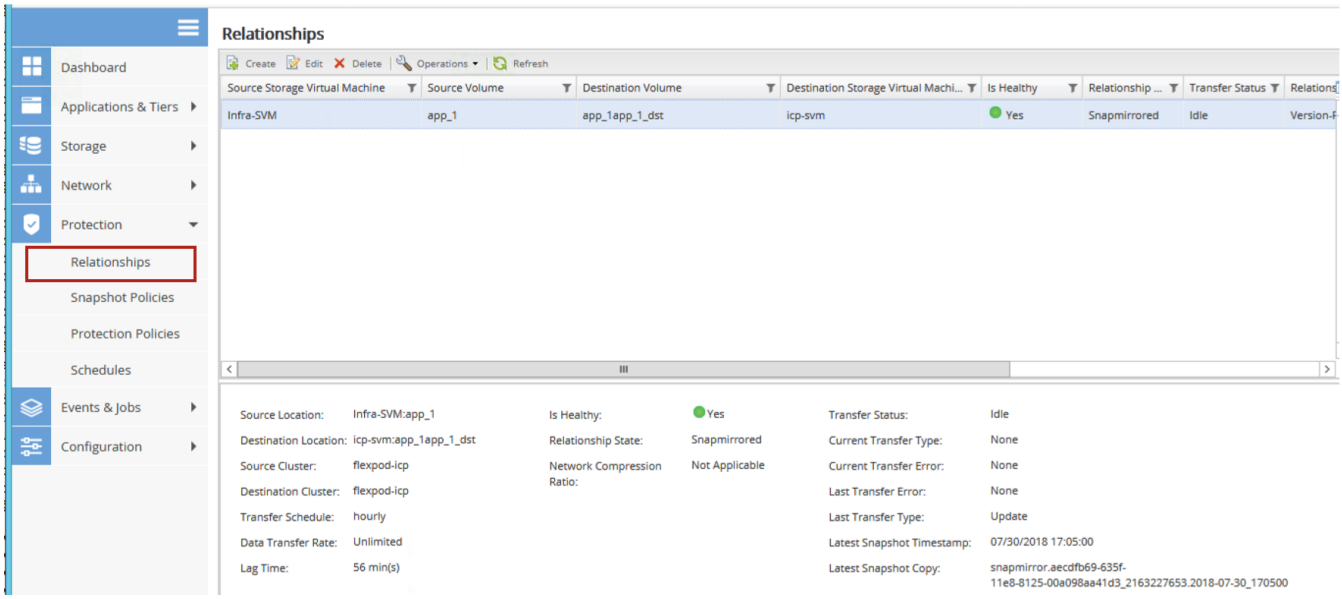
Data Protection was enabled to demonstrate the usage of Snapshots Policies as it was used in one of the storage classes that Trident can provision. The other data protection aspects configured is SnapMirror relationship to demonstrate the ability of the storage system to replicate data between different storage systems or between different SVMs even within the same storage cluster.

For data replication (SnapMirror) to work, relationship between the source and destination need to be configured and initiated. OnCommand System Manager has a simple wizard that simplifies the implementation of data replication using SnapMirror. Our source volume was app_1 on Infra-SVM and the destination volume was app_1app_1_dst on SVM icp-svm. As indicate in the details, the replication scheduled is hourly.

We used volume app_1 as it was the underlying storage volume for datastore applications_1 that was designated to host VMs with enterprise applications.

Figure 45 captures the state of the data protection relationship we created.

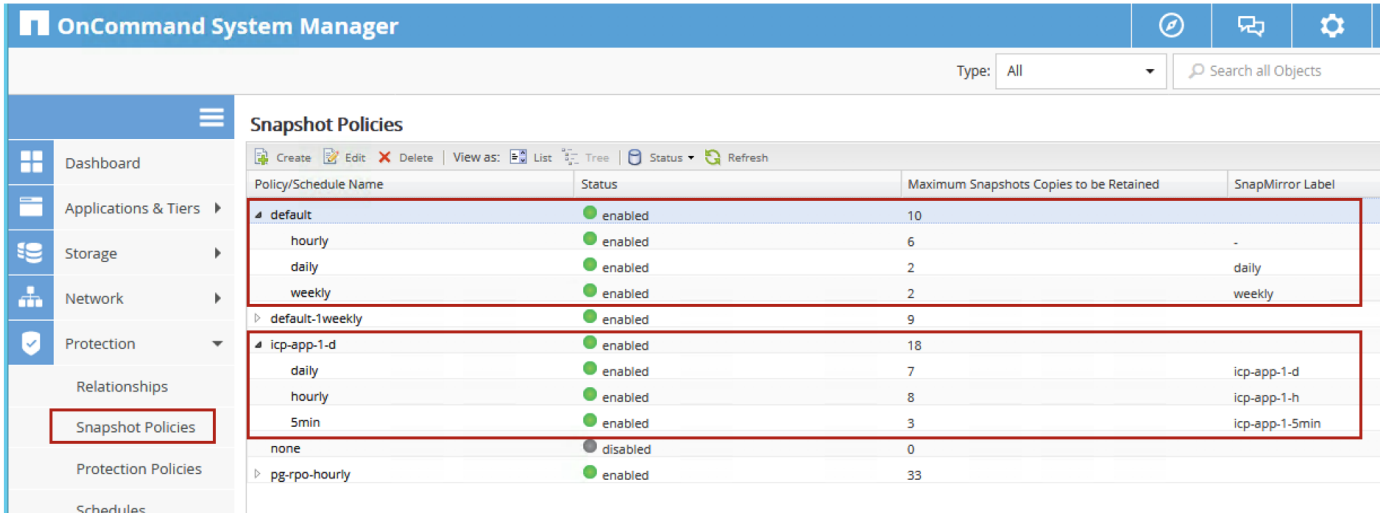
Figure 45 Data Protection - SnapMirror Replication



In our test lab, Data Protection policy with SnapMirror replication was implemented within the same ONTAP cluster in our A300 storage, but data replication can be implemented between different storage clusters that are part of FlexPods that reside in different data centers for DR purposes. Data replication can be used also in conjunction with cloning, using NetApp FlexClone. Data in the destination site can be cloned to enable DevOps related operations so the DR site can also act as another development and test environment. In addition, FlexClone can be utilized to test DR readiness while the data replication occurs in the background, minimizing business exposure to risks associated with data loss.

Snapshots are built-in data protection feature of the ONTAP storage OS. Figure 46 provides details about the Snapshot policies that were created. These snapshots are available also to be consumed by Trident as an attribute of the provisioned storage class. Creation of Snapshots policies can be done via the OnCommand System Manager UI not by Trident. In the .yaml file for a storage class, a specific snapshot policy can be added to the definition of the storage class. In the storage class `standard` we have been including the snapshot policy `default`.

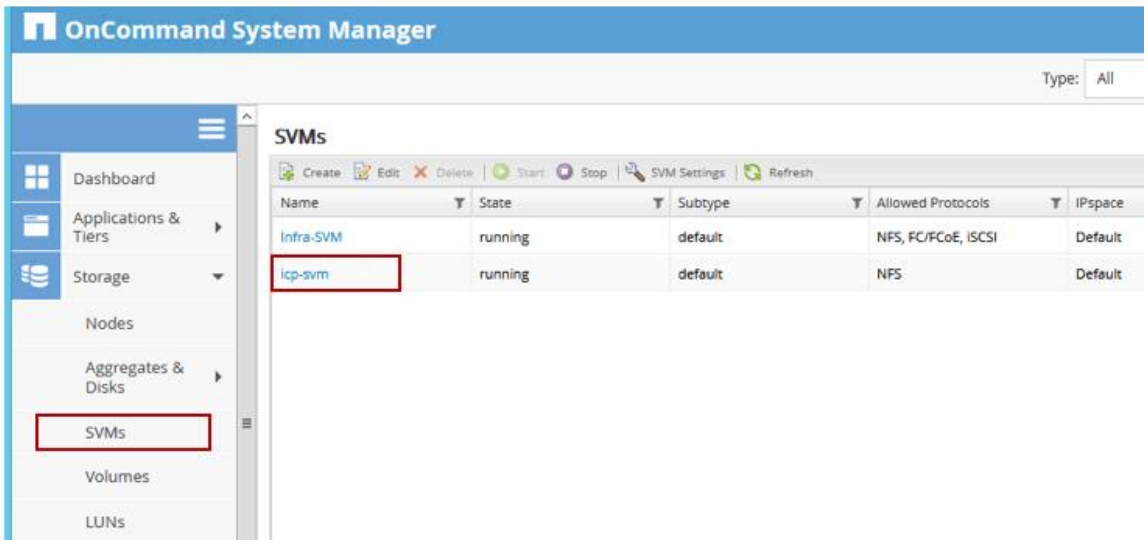
Figure 46 Snapshot Policies



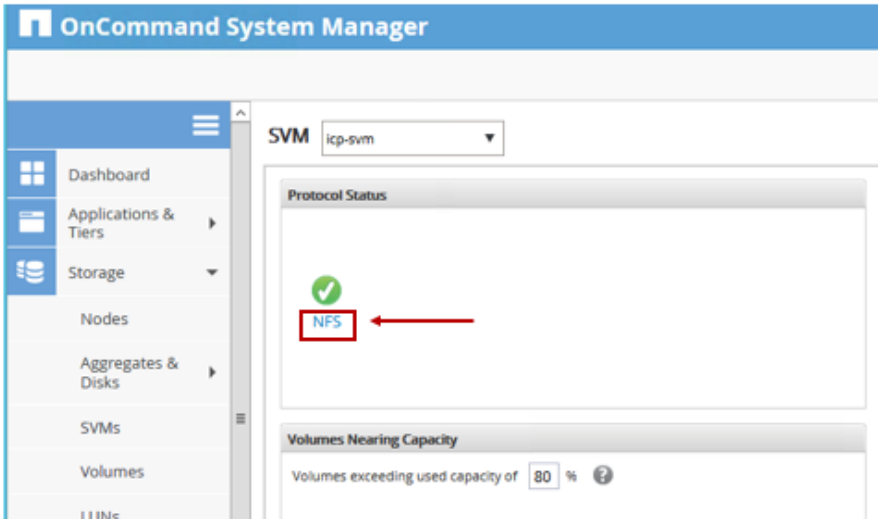
NFS Exports

As described in the Solution Design section, NFS storage services are key to the solution. Export policies enable access control. To define the export policies from the management UI under the associated SVM complete the following steps:

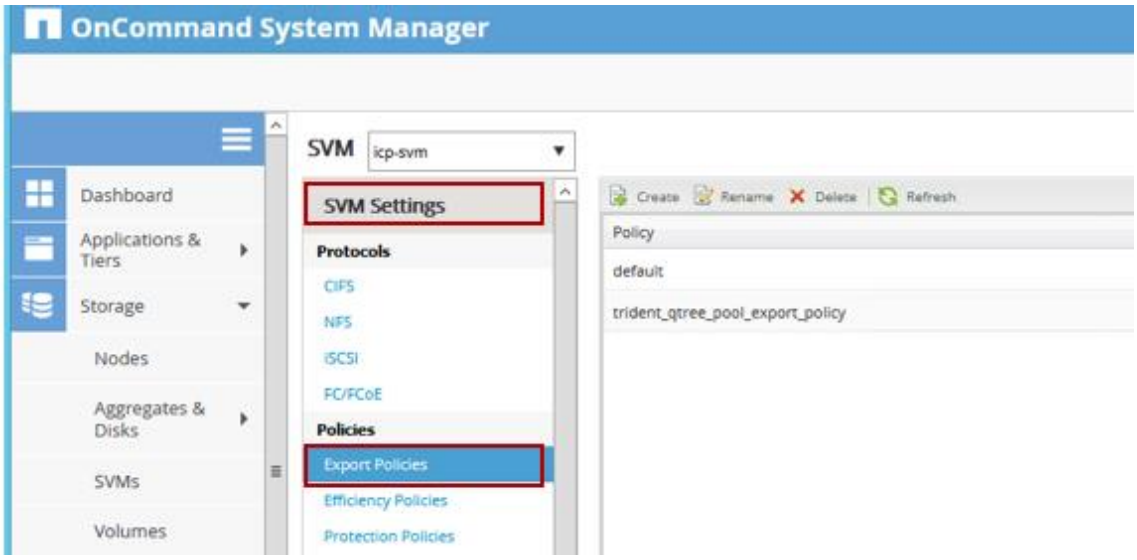
1. Under SVM, select the SVM in which the export policy will be created (`icp-svm` in our case):



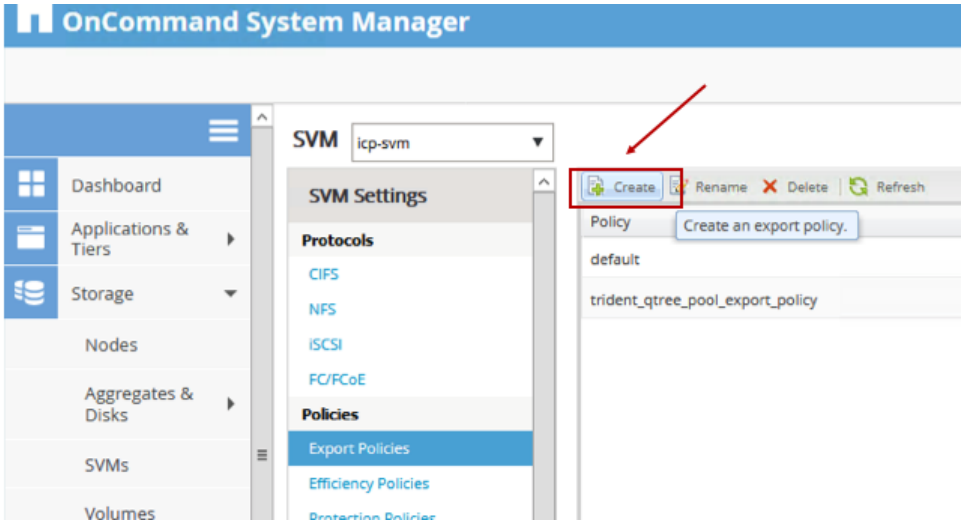
2. Select the NFS protocol listed.



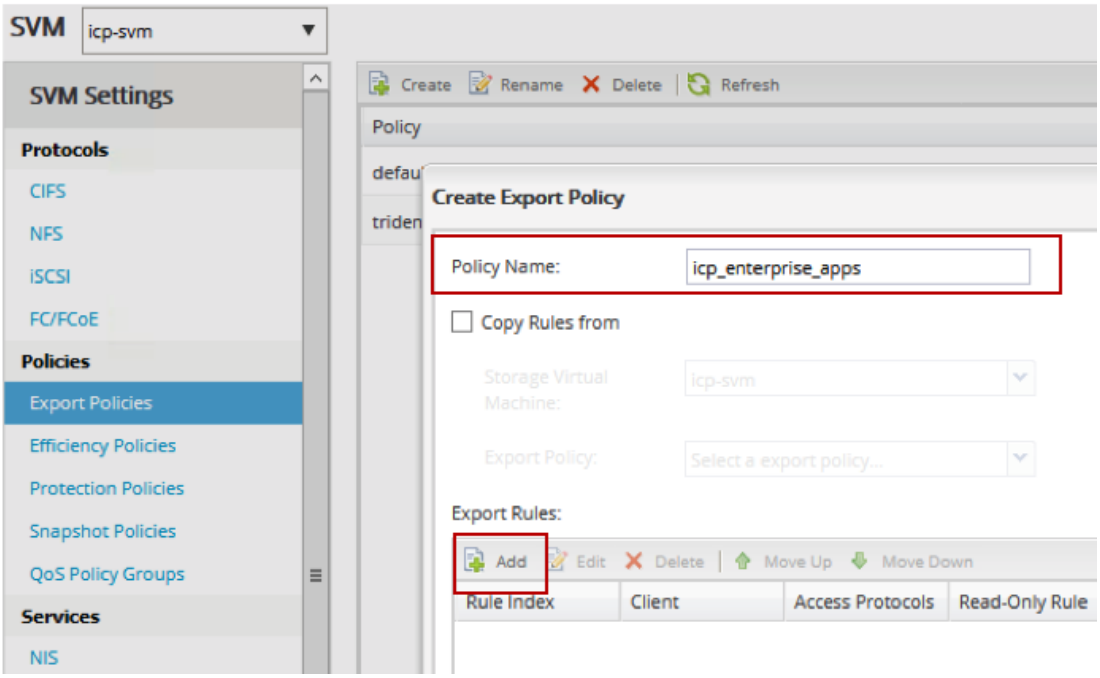
3. A new menu with SVM Settings will be revealed. Click the **Export Policies** option under Policies. In our case we already have two export policies listed: `default` and `trident_qtree_pool_export_policy`. We will create a third policy.



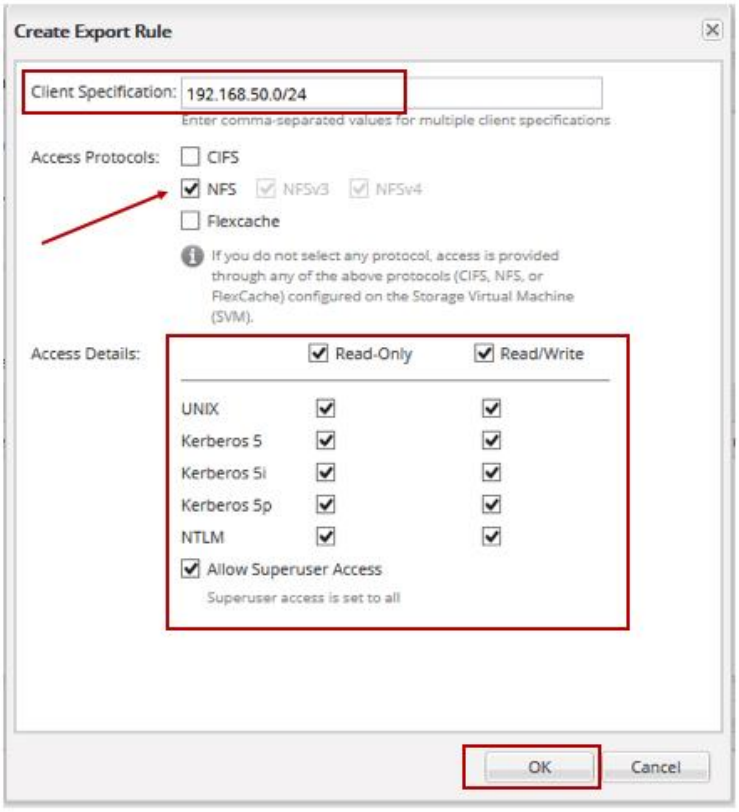
4. Click **Create** to start creating a new export policy



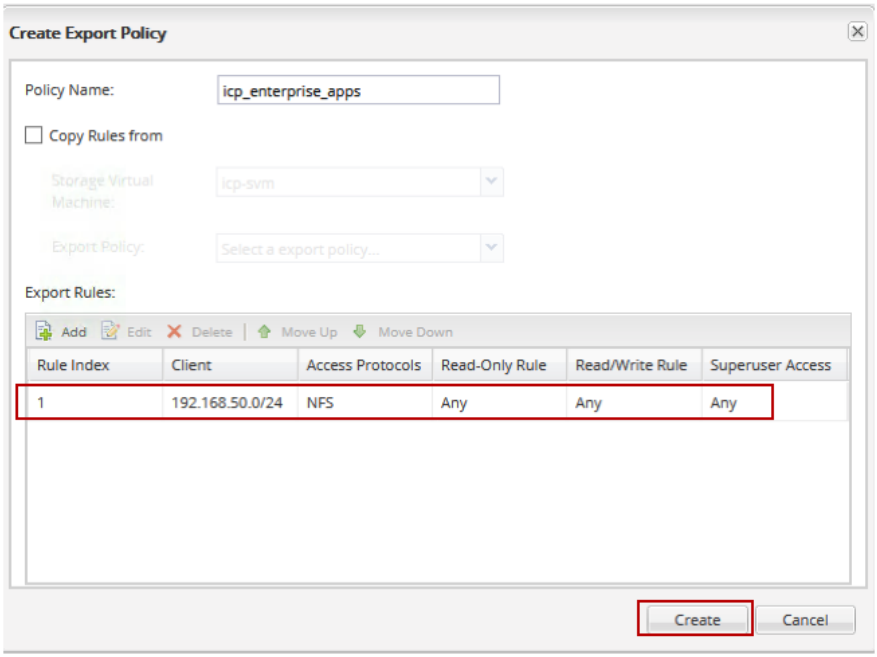
5. Name the export policy (`icp_enterprise_apps` used in our case), then click **Add** under the **Export Rules**



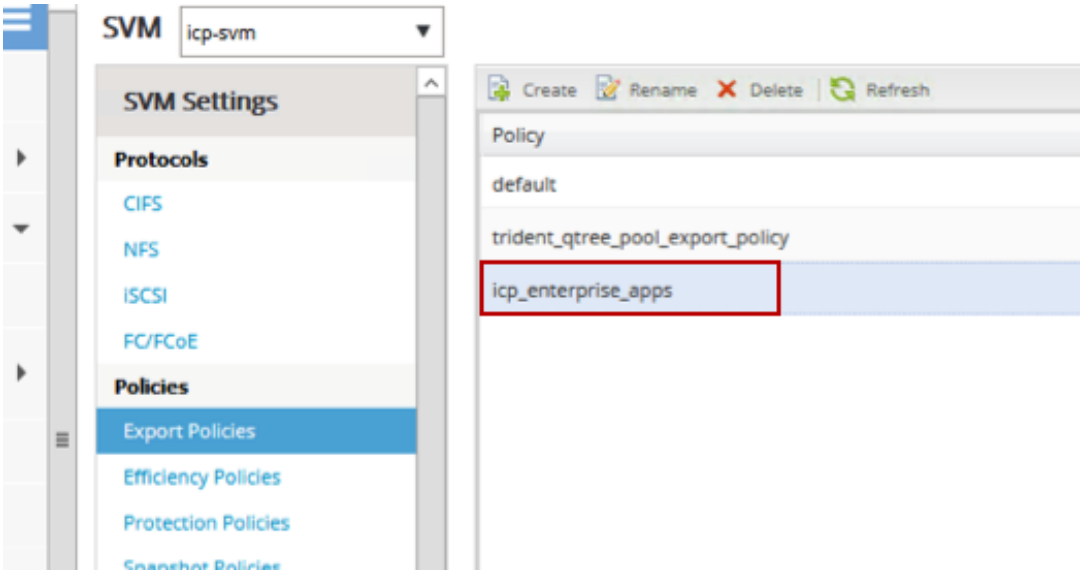
6. In the **Create Export Rule** window enter the client specification. We used the subnet of the ICP nodes. Also, check the **NFS** protocol and check the relevant **Access Details** options. When done click **OK**.



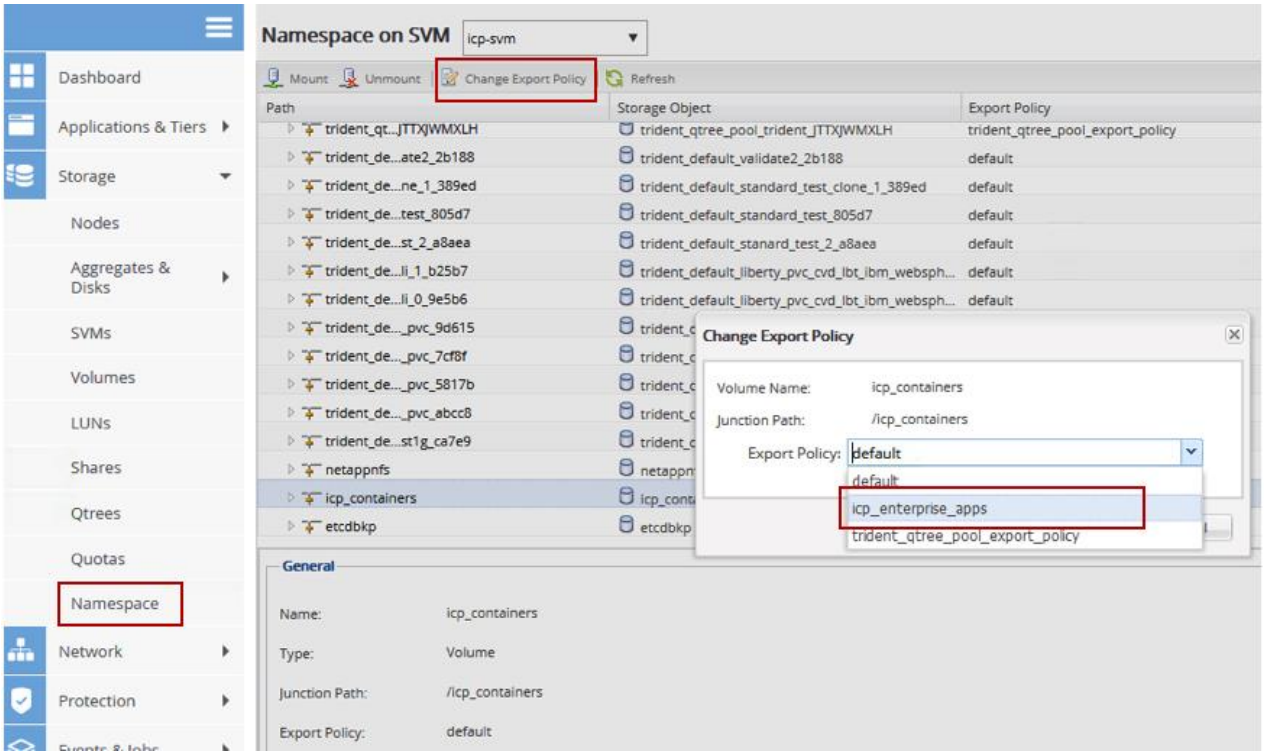
7. The new rule will be listed, then click Create to add the new policy to the SVM.



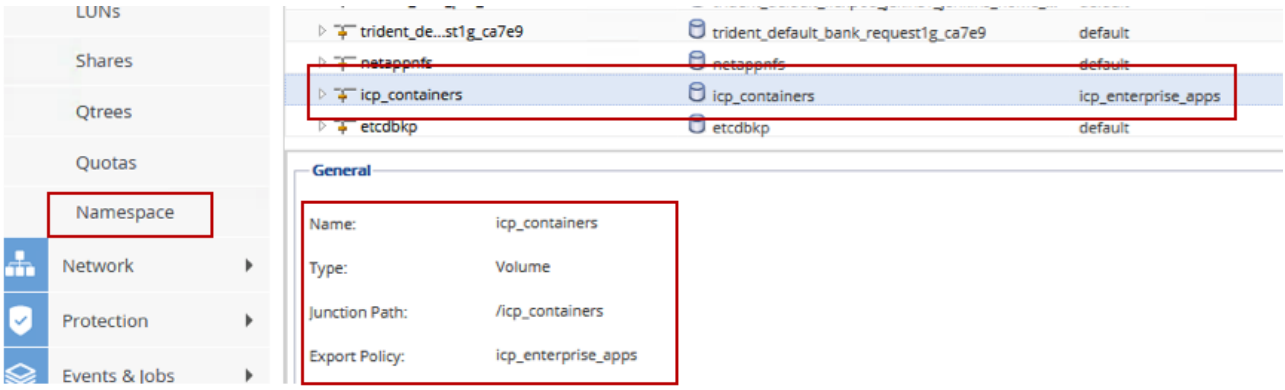
8. The new policy is now available and can be associated with the NFS exports.



9. You can now apply the export policy to the NFS volumes, which upon creation get the **default** export policy. In this case, apply the newly created policy to the volume created named `icp_containers`. Do so under the **Namespace** option in the Storage menu; by default, the junction path gets the name of the volume.



10. The volume now has the new export policy. It can be mounted at the node using the junction path `/icp_containers`.



FlexPod Network Configuration

The initial setup and network configurations of the FlexPod system are beyond the scope of this document and are covered in a previously published CVD; refer to the FlexPod Design Guide for more information.

Create a New VMware Port Group for ICP environment

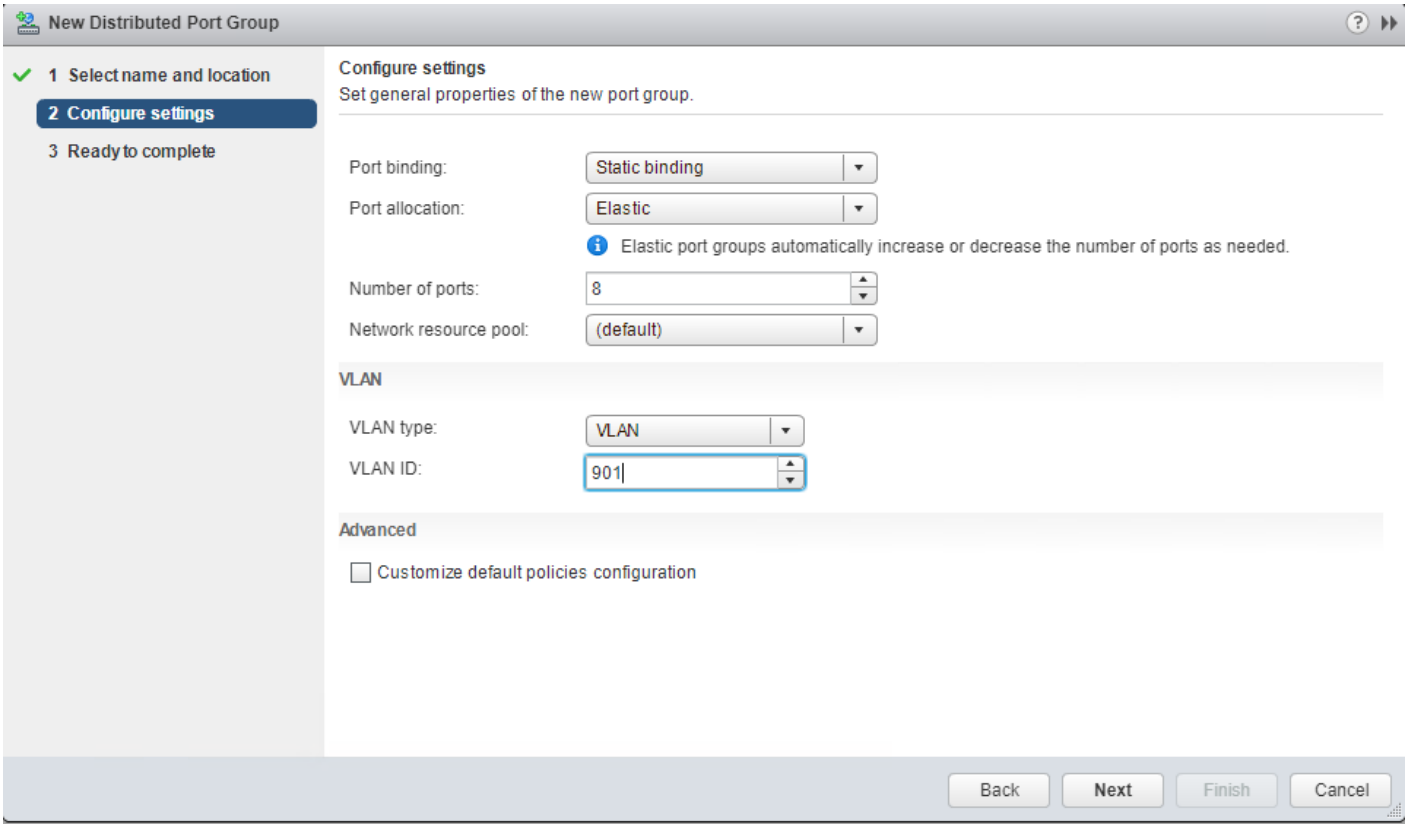
To create two new distributed port groups on the FlexPod application (VDS) switch to support ICP management and NFS traffic, complete the following steps:

1. Login to VMware vCenter web client and select the **Networking** inventory view.

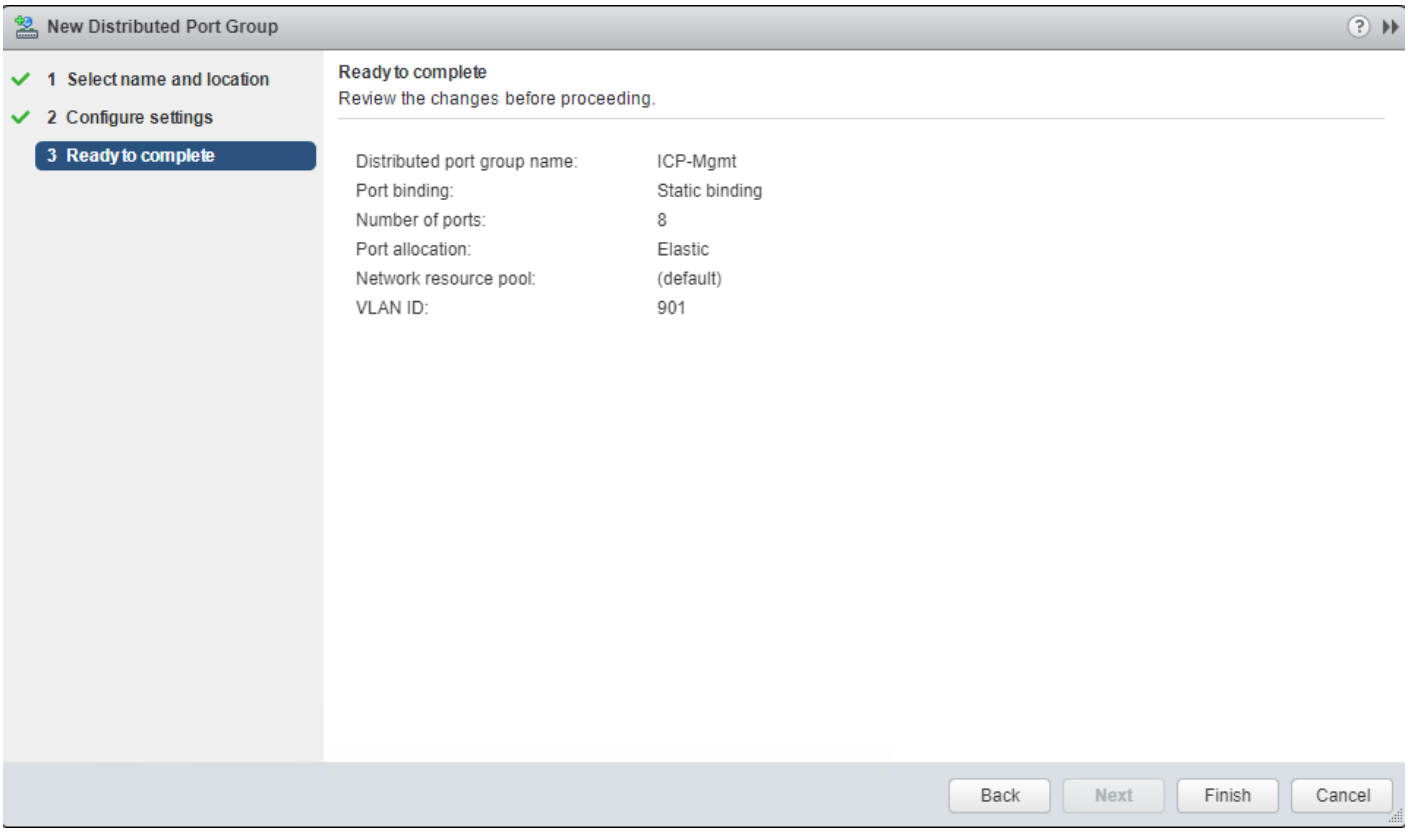
2. Select Inventory **>** vSphere Distributed Switch **>** New Port Group.

3. Enter a Name (ICP-DportGroup) for the new distributed port group.

4. Select VLAN type and VLAN id for the ICP traffic.



5. Click **Next** and click **Finish.**



6. Repeat the steps mentioned above to create another port group for NFS traffic.

FlexPod VMware vSphere Configuration

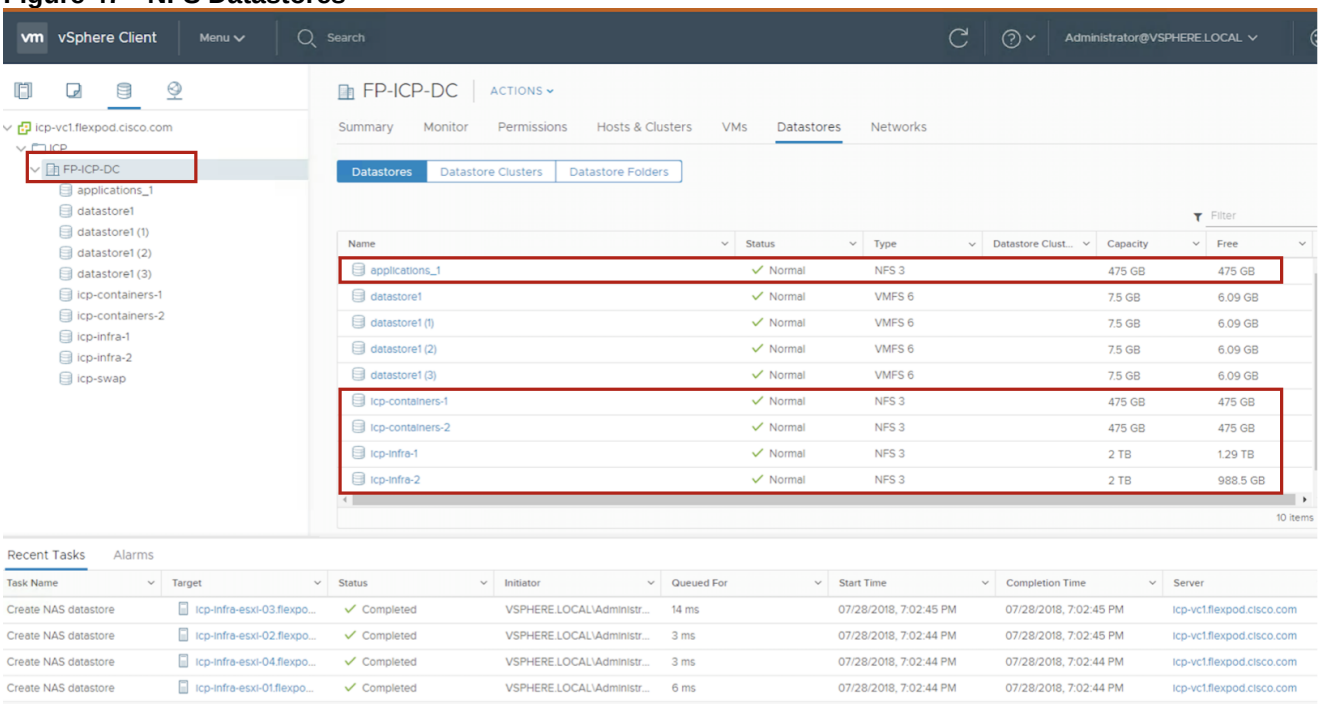
Most of the vSphere related settings are covered in a previously published CVD and other technical documents published by NetApp and Cisco. This section does provide some details about the configuration settings of the datastores, the networks, and the DRS rules used in our ICP test environment.

NFS Datastores

NFS datastores are used as the underlying storage for the ICP nodes. The datastores are highly available at the storage level, protected by the HA pair of the two-node A300 cluster and are available to all ESXi hosts. Several datastores were configured but the total number will vary depending on the number of nodes and applications, and the business requirements in terms of data protection policies and other data management and operational practices. In the ONTAP storage, most of the data management features and policies are applied at the volume level, which is presented to vSphere for the creation of the datastore. It will be practical and efficient to use the same datastore for VMs that share the same set of data management requirements such as data replication or snapshots policies.

Five NFS-based datastores were configured and tested in our environment, as shown in Figure 47. The datastores were associated with all four ESXi servers to enable highly available access to the underlying storage the VMs are deployed on and to support continuous operation in case of a vMotion operation when a VM is moved to a different ESXi host. Five NFS-based Datastores are listed in our test environment: `applications_1`, `icp-containers-1`, `icp-containers-2` and `icp-infra-1` and `icp-infra-2`, with the last two used for hosting the ICP nodes.

Figure 47 NFS Datastores



Networks

The vSphere environment was configured with the following network settings: `ICP-NFS-Data` network was created and is associated with all four hosts. A distributed Switch, `DSwitch-ICP` with two port groups: `DPortGroup-ICP-Mgmt` and `DPortGroup-NFS-ICP` for managing iSCSI and NFS traffic respectively, and Uplink Port Group: `DSwitch-ICP-DVuplink-22` was created. Your IT best practices as well as recommendations from VMware, NetApp and Cisco should be followed to meet the specific design for your environment. The settings are captured in Figure 48 through Figure 51.

Figure 48 Networks

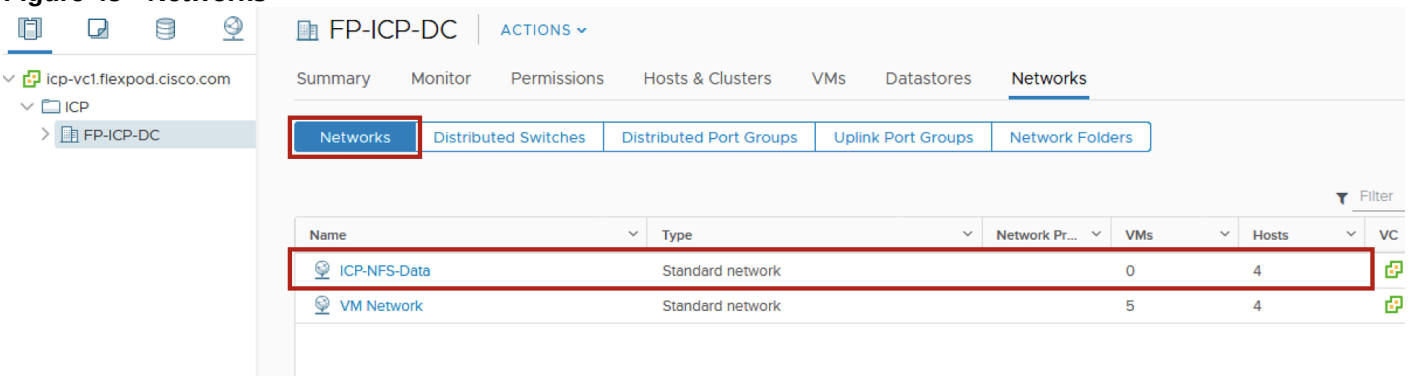


Figure 49 Distributed Switch

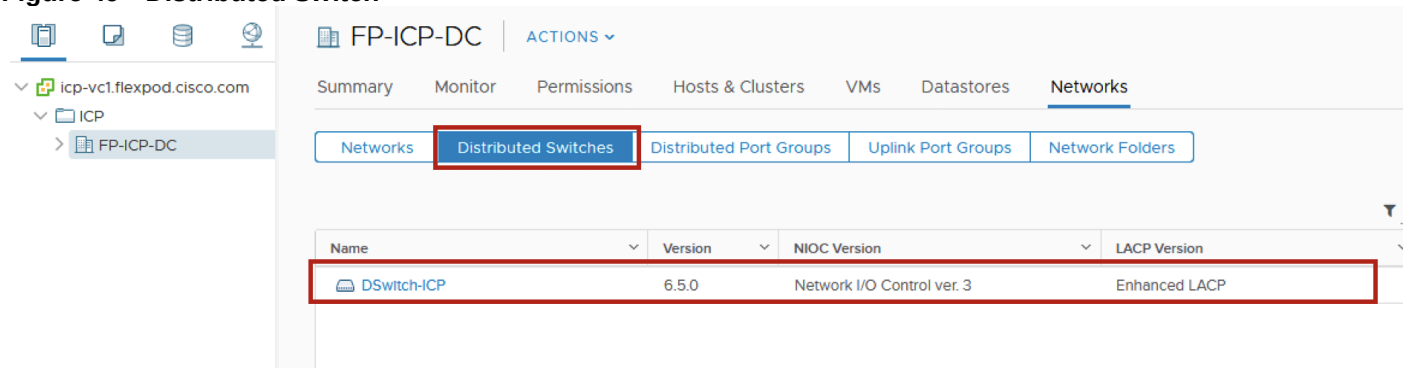


Figure 50 Port Groups

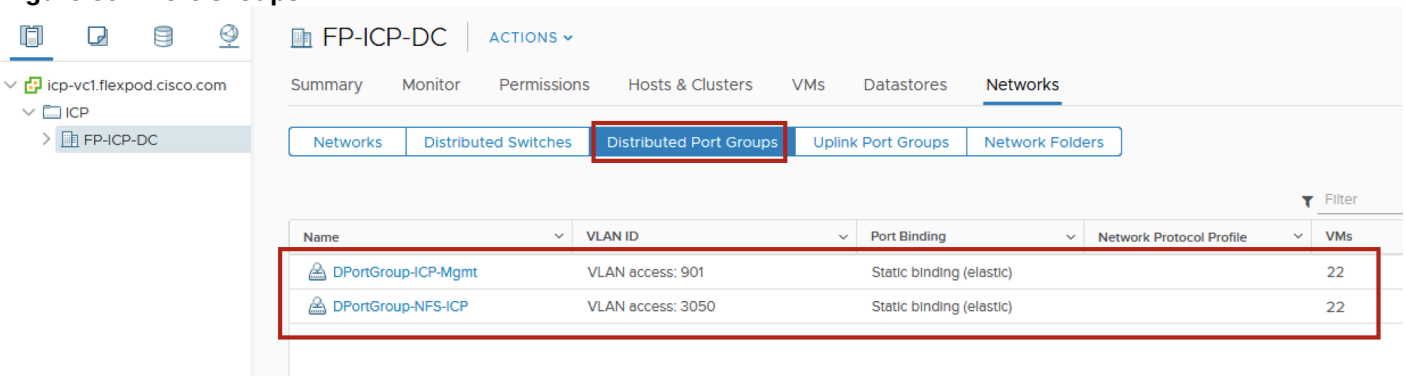
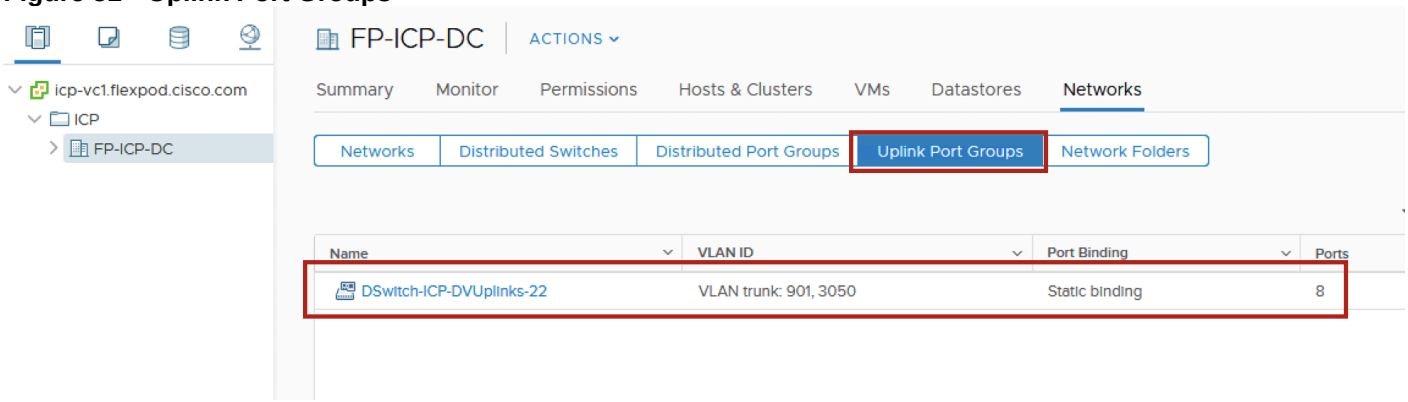


Figure 51 Uplink Port Groups



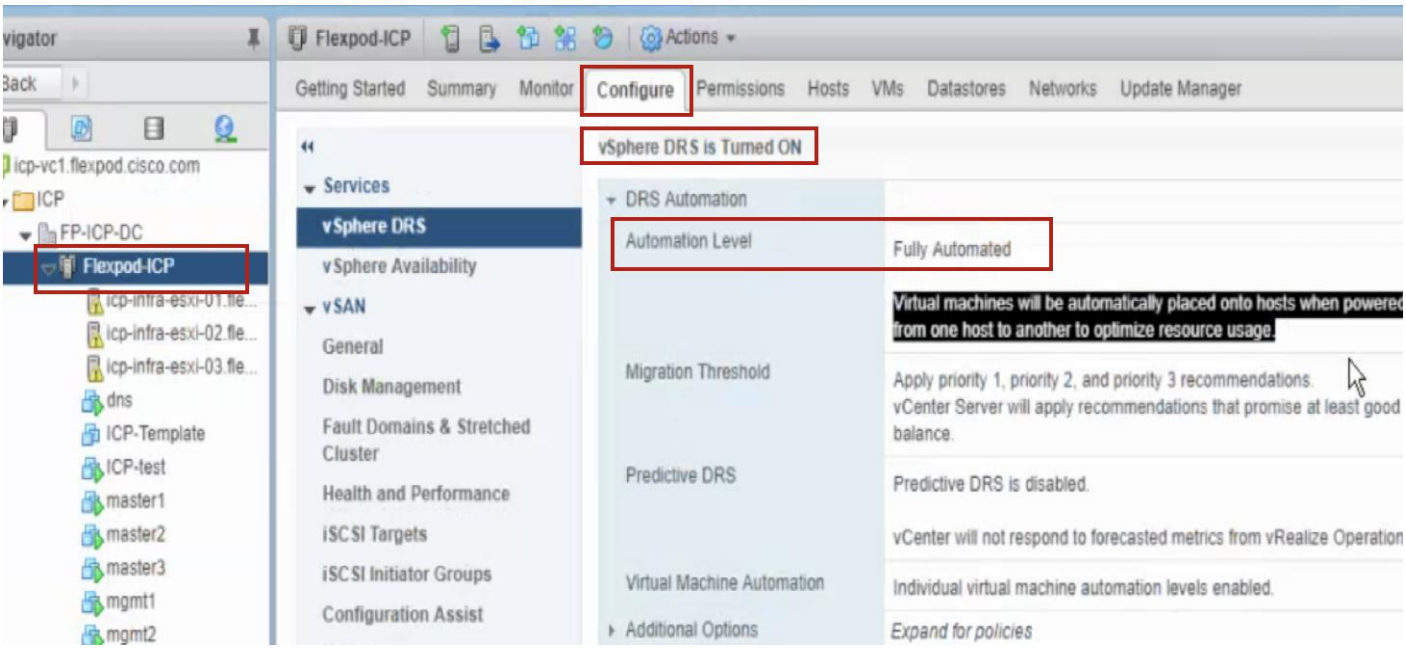
Setting Up vSphere Distributed Resource Scheduler (DRS) Rules

In order for the ICP cluster to benefits from the underlying resiliency capabilities of the vSphere layer, DRS rules need to be defined and vMotion enabled as explained in the Design section. In our particular configuration we wanted to ensure that not more than one Master node will be hosted on an ESXi server. Similar rule can be set for the Management node.

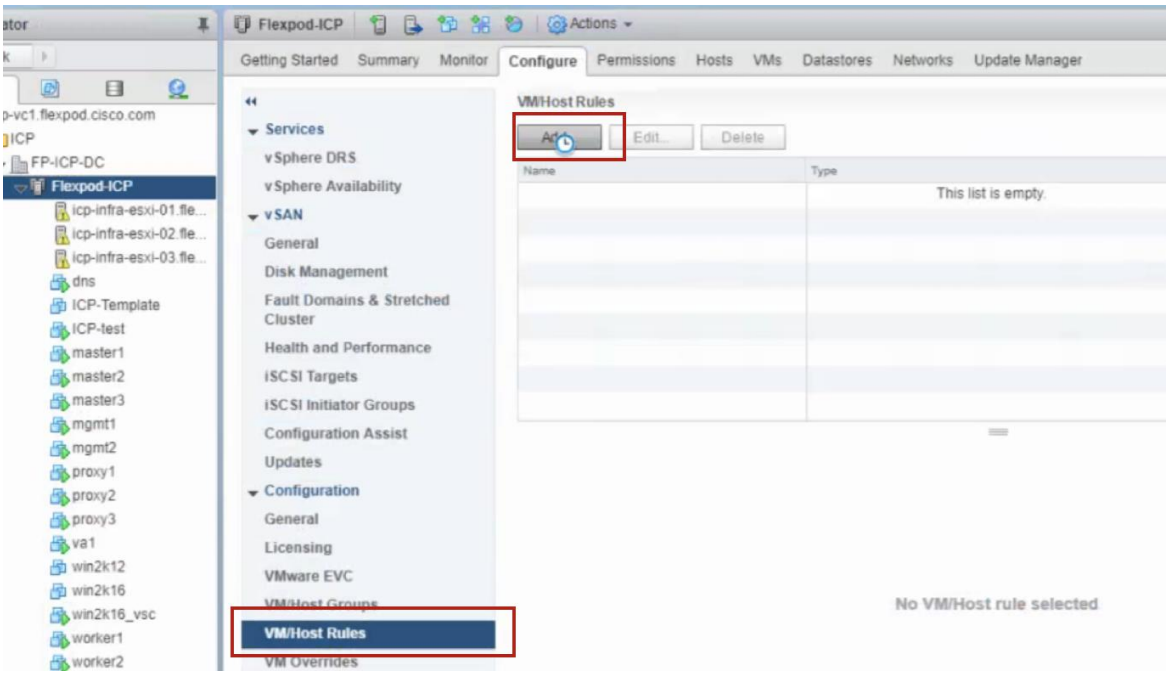
Since the solution requires three Master nodes, it also means that at least three ESXi hosts are required, regardless of the required compute capacity. If we want to further protect the environment from a failure of one ESXi server, then a fourth ESXi needs to be included in the solution, as we have implemented in our test environment.

DRS and vMotion can be configured from the vCenter Configure tab for the cluster after the completion of the deployment of all the VMs that participate in the ICP cluster or at least those VMs that will be members in the DRS rule. It can also be configured upon the completion of the ICP installation. In either case it is recommended to test and confirm that the rules are properly in place.

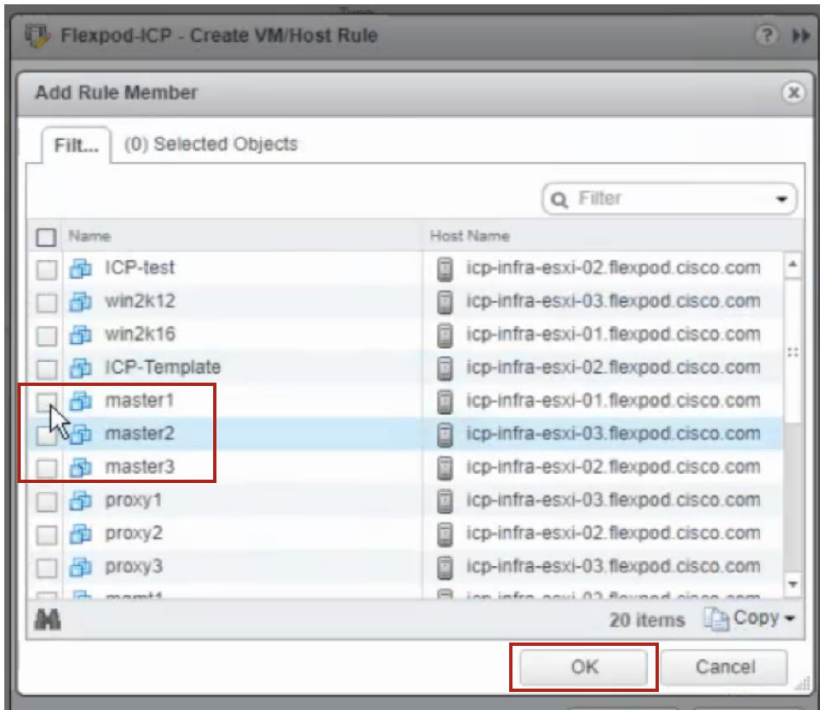
1. From the **Flexpod-ICP** cluster in the **Configure** option make sure that DRS is turned ON and should show as Fully Automated.



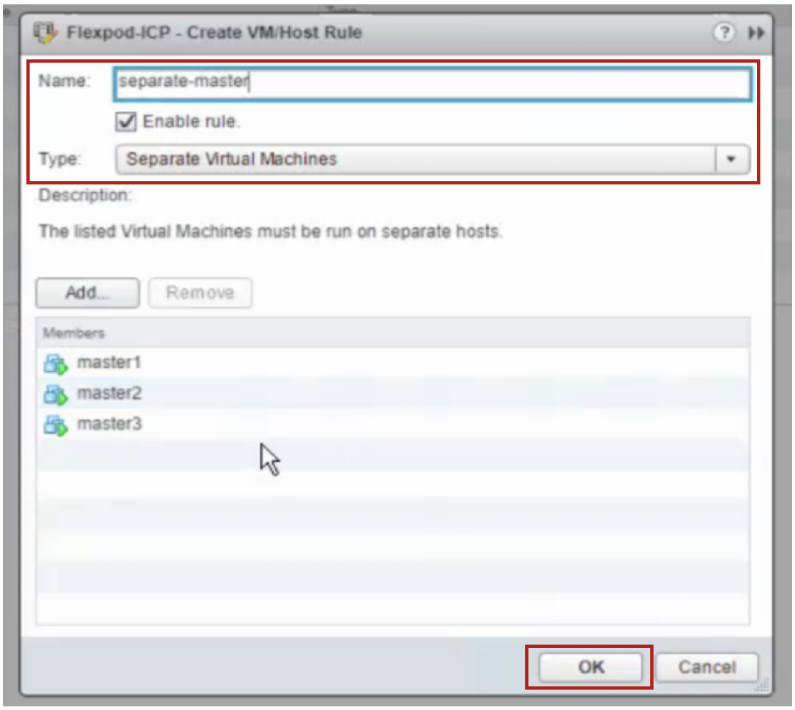
2. Click the **VM/Host Rules** under **Configurations**, then click **Add** to add a new rule.



3. Select the three Master servers as members of the new rule then click **OK**.

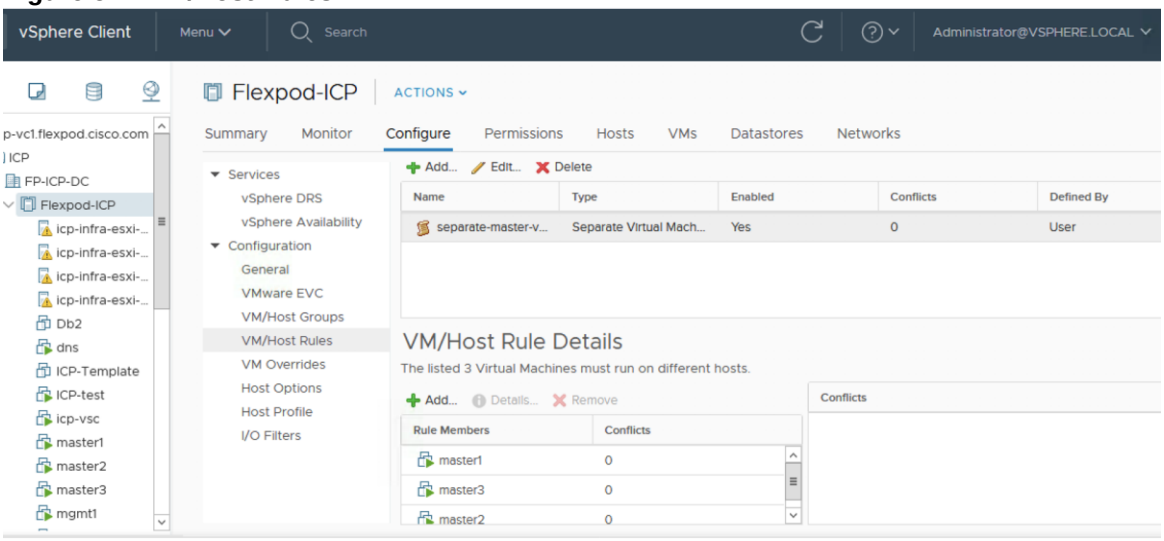


4. Give the rule a meaningful name, make sure the **Enable rule** is checked and the type is **Separate Virtual Machines**, the click OK.



The rule is now in place and will be enforced. The desired behavior was validated during testing.

Figure 52 VM/Host Rules

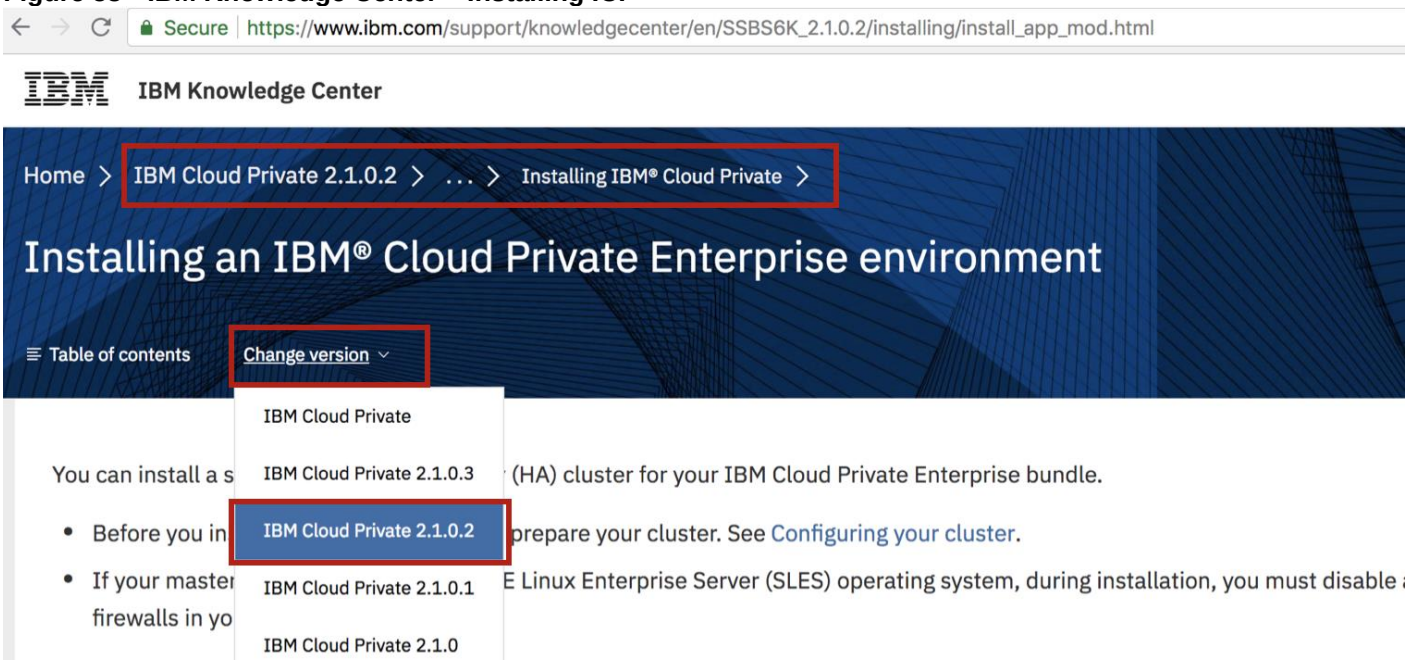


IBM Cloud Private Installation and Configuration

The installation of the ICP platform is a multi-steps process that has to be carefully planned for each specific environment. There are several prerequisites for preparing the infrastructure and the nodes for the installation, and it is also important to recognize that the installation procedure may vary between the different versions of ICP.

It is recommended to check the installation procedures published by IBM and match them to the specific release of ICP. See https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.02/installing/ and Figure 53.

Figure 53 IBM Knowledge Center – Installing ICP



Cloud-Native and the Enterprise editions of ICP include the same ICP core packaged software. ICP Enterprise Edition includes additional software packages of containerized IBM applications that have to be obtained separately and brought into the ICP catalog as Helm charts after ICP has been installed so they can be deployed.

IBM also offers a version of ICP that is based on Cloud Foundry, a version we have not tested or qualified in this CVD.

Prerequisites and Preparation Stage

As a reminder and as covered in the Solution Design section, we are deploying ICP version 2.1.02 with 14 nodes (Table 2). IBM has simplified the installation process in a way that once the infrastructure and related aspects are configured and ready, one Ansible script needs to be executed to handle the actual installation of the ICP software.

Infrastructure

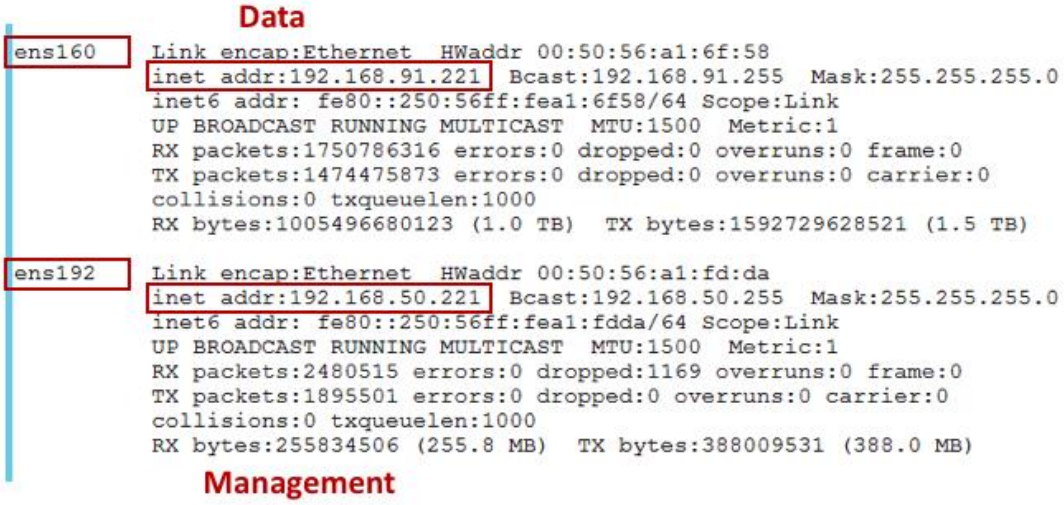
Infrastructure components and related aspects must be deployed and configured according to best practices covered in previously published CVDs and other best practices from Ubuntu, VMware, Cisco, NetApp and IBM, and are not covered in detail in this document.

The following is a short description of these infrastructure components and aspects that must be addressed prior to installing ICP:

1. Storage, network and servers were deployed, connected and configured in HA settings to meet the requirements and were tested
2. ESXi hosts were deployed and configured under the designated data center and ICP cluster within an existing vCenter 6.5
3. DNS and ADS available and ready
4. NFS datastores were configured and associated with all ESXi hosts
5. vMotion was enabled

6. Storage specific:
   a. Storage aggregates were created
   b. LIFs were created, IP addresses assigned
   c. SVMs were created and configured
   d. Export policies were created
   e. Data Protection policies (data replication as well as snapshots policies) were configured as desired.
   f. Quality of Service (QoS) policies were created
   g. Storage efficiency features were enabled
7. IP addresses for nodes VMs
8. The ICP nodes were deployed with each Ubuntu VM connected to two networks: management and data. 192.168.91/24 for data and 192.168.156/24 for management. Static IPs were designated to each node with additional IPs designated to virtual IPs for accessing the cluster. Interface `ens160` was used for data and interface `ens192` for management. Figure 54 shows the interfaces in the master1 node VM, and all other node VMs were configured in the same way.

Figure 54 Network Interfaces on the Ubuntu VMs



9. VM resources
   Planning for compute resources is a required step as part of sizing the infrastructure for the ICP cluster. The UCS B-series M5 servers validated in the solution support broad range of Intel Xeon processors and can be loaded with up to 3TB of RAM, offering ample compute power to host significant workloads. Additional important sizing considerations are provided in the Solution Design section of this document. Table 3 in the Solution Design section provides the specification of each node VM according to the environment implemented for this CVD.
10. Create a VM template
    All the VMs for the ICP nodes were deployed with Ubuntu 16.04 as the guest OS on the ESXi. Since we will be deploying 14 VMs and more may be added in the future to support growth, it is recommended to create a template that can then be cloned and then modified for the specific node with the proper allocation of resources and IP address. OS should be updated and hardened according to the best practices that were adopted by the IT organization. SSH client needs to be included in the template or installed after a VM is deployed and the same for NFS services: `apt-get update` then `apt-get install nfs-common`.

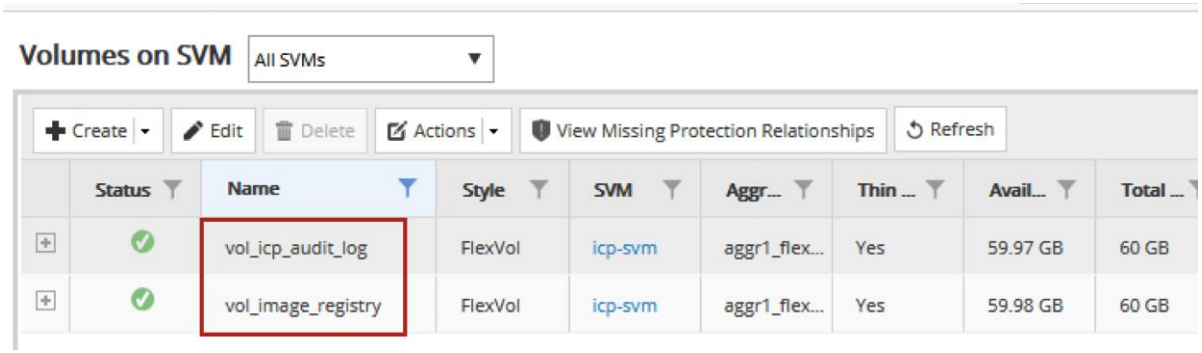
11. Deploy all nodes from the created template. Ensure all nodes are up-to-date, secured and that each node is accessible from all other nodes and clocks are synchronized.

12. Firewall ports are opened to support ICP, see:

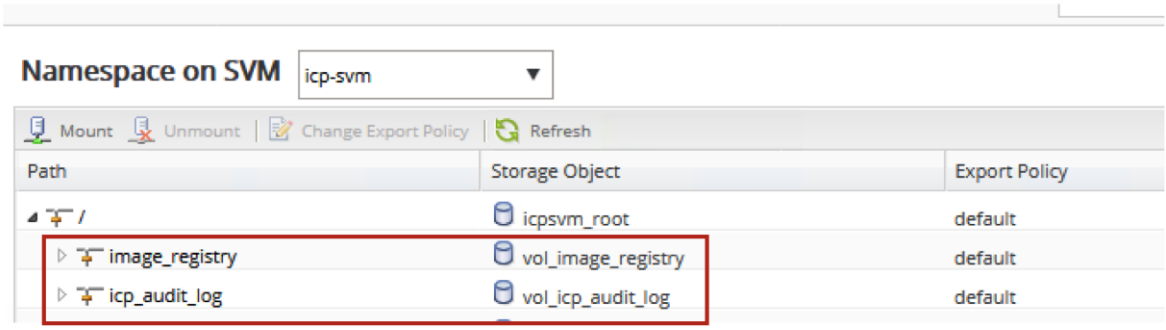
https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/supported_system_config/required_ports.html

To create shared NFS for key ICP files, complete the following steps:

1. Create two 60GB volumes on the storage as shown below, following the steps provided earlier for NFS exports. We named the volumes: vol_icp_audit_log and vol_image_registry. The volumes will be used to store the private image registry and the ICP audit log. The shared storage provides access to multiple nodes at the same time, it is highly available and it can be grown easily if additional capacity is needed. These volumes will be mounted to the three Master nodes.



2. Check in the Namespace menu option and find the Junction Path for the volumes (modify path if preferred). The Junction Path will be used in the three Master nodes for mounting the volumes. The two paths listed are: image_registry and icp_audit_log.



3. On each of the three Master nodes, mount the two volumes as shown below, making sure they are permanently mounted by adding them to the /etc/fstab.

```
#NFS mounts for ICP
nfs1:/image_registry  /var/lib/registry nfs defaults 0 0
nfs2:/icp_audit_log       /var/lib/icp/audit nfs defaults 0 0
```

Additional NFS exports can be created for other files as needed (for example, Docker repository). For more information see:

https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/supported_system_config/hardware_reqs.html

Preparing the ICP Cluster

To prepare the ICP Cluster, complete the following steps:

1. Add IP addresses and host names to etc/hosts files in all nodes (see Figure 55), also, comments out the local host 127.0.0.1

Figure 55 IP Addresses and Host Names in the etc/hosts File

```
192.168.91.221  master1
192.168.91.222  master2
192.168.91.223  master3
192.168.91.224  proxy1
192.168.91.225  proxy2
192.168.91.226  proxy3
192.168.91.227  mgmt1
192.168.91.228  mgmt2
192.168.91.229  worker1
192.168.91.230  worker2
192.168.91.231  worker3
192.168.91.232  worker4
192.168.91.233  worker5
192.168.91.234  va1
192.168.91.235  cluster_vip
192.168.91.236  proxy_vip
192.168.91.237  dns
192.168.91.238  worker6
```

2. Python 2.6 – 2.9 is required as it will be used by Ansible to deploy the ICP software across all nodes. Install Python if not already installed and confirm. If not available, Python 2 can be installed by running: `apt install python-minimal`

```
root@worker1:/# python --version
Python 2.7.12
root@worker1:/#
```

3. On all three master nodes you need to make sure that the `vm.max_map_count` setting is at least `262144`.We do so by ruining `sysctl -w vm.max_map_count=262144`. To maintain the settings permanently, update the `/etc/systctl.conf` file to include this setting.

```
#
# Log Martian Packets
#net.ipv4.conf.all.log_martians = 1
#
vm.max_map_count=262144
net.ipv4.ip_local_port_range="10240 60999"
net.ipv4.ip_forward=1
root@master1:/#
```

4. Additional setting on the three master nodes is to ensure that the ephemeral port range is greater than 10240. To make the change we run `sysctl -w net.ipv4.ip_local_port_range="10240 60999"`. To keep the settings permanently, we need to edit the `/etc/systctl.conf` file.

```
#net.ipv4.conf.all.log_martians = 1
#
vm.max_map_count=262144
net.ipv4.ip_local_port_range="10240 60999"
net.ipv4.ip_forward=1
root@master1:/#
```

5. The installation of ICP is done from the boot node, which in our case is a component of the master1 node and not a separate VM. Docker needs to be installed on the Boot node since the installation package comes as a Docker container. Please make sure you install a version of Docker that is supported by IBM, see: https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/supported_system_config/supported_docker.html

Docker 17.09 (CE) was installed in our environment, first on the master1 node and in a later step on the other nodes as well. Docker must be enabled to start at boot.

```
root@master1:/opt/ibm-cloud-private-2.1.0.2/cluster# docker -v
Docker version 17.09.1-ce, build 19e2cf6
root@master1:/opt/ibm-cloud-private-2.1.0.2/cluster#
```

Help installing Docker can be found here: https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/installing/install_docker.html#verify

6. Share the SSH key. Generate the key from the Boot node by running:
   `ssh-keygen -b 4096 -f ~/.ssh/id_rsa -N ""`

7. Add the key to the authorized keys and then add the key to all the nodes in the cluster. See:

   https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/installing/ssh_keys.html

8. SSH services need to be restarted on all nodes:
   `systemctl restart sshd`

9. Obtain the ICP installation file and download it to a temporary location.
   Make sure you obtain the correct installation file: `ibm-cloud-private-x86_64-2.1.0.2.tar.gz` or `ibm-cp-app-mod-x86_64-2.1.0.2.tar.gz`
   The file can usually be obtained from IBM's Passport Advantage site. See https://www-01.ibm.com/software/passportadvantage/

10. Extract the image file and load them into Docker, which should already be installed.

    ```
    tar xf ibm-cloud-private-x86_64-2.1.0.2.tar.gz -0 | sudo docker load
    ```

11. Create an installation directory for the CIP files and change into this directory.

    ```
    mkdir /opt/ibm-cloud-rpiavte-2.1.0.2
    ```

    then

    ```
    cd /opt/ibm-cloud-private-2.1.0.2
    ```

12. Extract the configuration file from the installer image.

    ```
    Sudo docker run -v $(pwd):/data -e LICENSE=accept ibmcom/icp-inception:2.1.0.2-ee
    cp -r cluster /data
    ```

13. A cluster directory is created in the installation directory. You will need to the hosts file the IP of each node in the cluster.

Figure 56 Hosts IP address

```
root@master1:/opt/ibm-cloud-private-2.1.0.2/cluster# cat hosts
[master]
192.168.91.221
192.168.91.222
192.168.91.223

[worker]
192.168.91.229
192.168.91.230
192.168.91.231
192.168.91.232
192.168.91.233
192.168.91.238

[proxy]
192.168.91.224
192.168.91.225
192.168.91.226

[management]
192.168.91.227
192.168.91.228

[va]
192.168.91.234
```

14. Replace the ssh_key in the cluster directory with the other private key file that is used to communicate with the other nodes in the cluster. `sudo cp ~/.ssh/id_rsa ./cluster/ssh_key`

15. Move the installation image to the images directory in the cluster directory prior to executing the installation script. Create the `images` directory under the `cluster` directory first, then move the file.

    ```
    mv /<path_to_installation_file>/ibm-cloud-private-x86_64- 2.1.0.tar.gz
    cluster/images/
    ```

```
root@master1:/opt/ibm-cloud-private-2.1.0.2/cluster/images# ls -l
total 8268648
-rw-r--r-- 1 root root 8467089796 May 30 16:53 ibm-cloud-private-x86_64-2.1.0.2.tar.gz
root@master1:/opt/ibm-cloud-private-2.1.0.2/cluster/images#
```

16. Add the High Availability settings to the `config.yaml` file (in the cluster directory).

```
## High Availability Settings for master nodes
vip_iface: ens160
cluster_vip: 192.168.91.235

## High Availability Settings for Proxy nodes
proxy_vip_iface: ens160
proxy_vip: 192.168.91.236
```

More information about this step can be found from the link below:
https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/installing/custom_install.html#HA

The `config.yaml` file can be modified to manage various aspects of the ICP installation. For more information, see: https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/installing/config_yaml.html

17. Prior to running the installation script we need to install Docker on all other nodes, or configure the cluster nodes for automatic Docker installation (done in the config.yaml file). For the Docker installation options, see: https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/installing/docker_cluster.html

ICP – Installation

To install ICP, complete the following steps:

1. From the cluster directory run the installation script to deploy ICP. This is an Ansible playbook put together by IBM:

```
sudo docker run --net=host -t -e LICENSE=accept -v "$(pwd)":/installer/cluster
ibmcom/icp-inception:2.1.0.2-ee install
```

```
root@master1:/opt/ibm-cloud-private-2.1.0.2/cluster# sudo docker run --net=host -t -e LICENSE=accept \
> -v "$(pwd)":/installer/cluster ibmcom/icp-inception:2.1.0.2-ee install
```

2. On completion of the installation, you should see a message with the access information. The config.yaml file has the default user name and password set as admin/admin

Figure 57 Message Indicating the Completion of the Installation

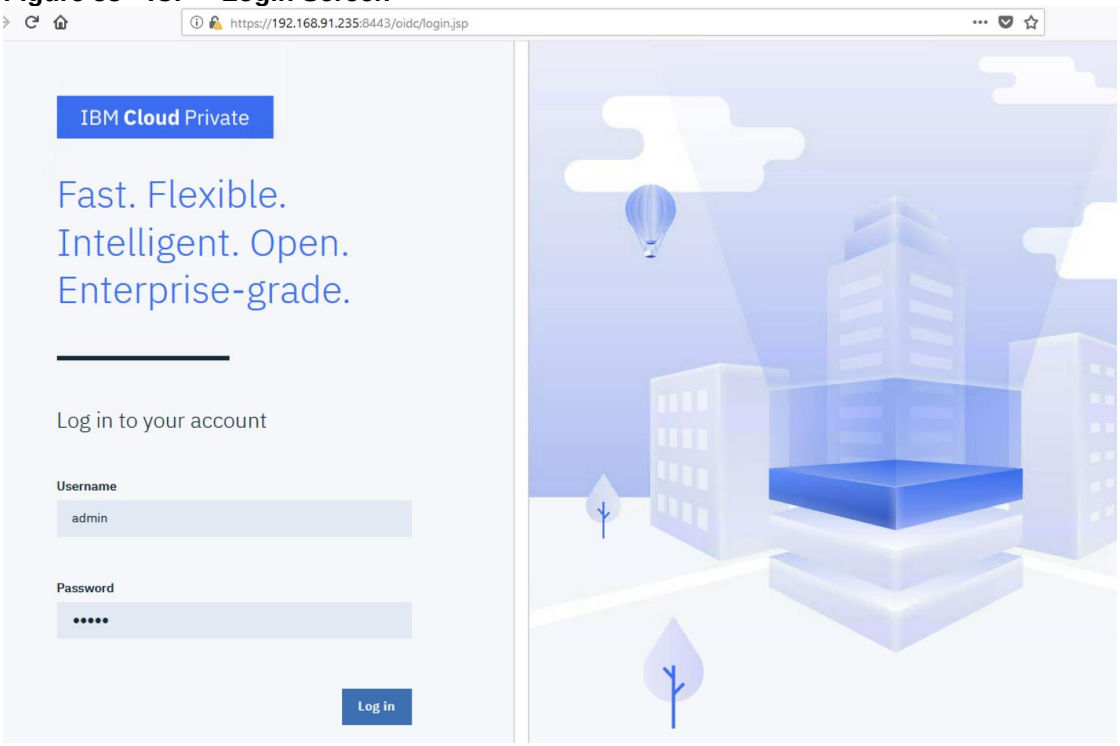
```
POST DEPLOY MESSAGE ***************************************************************************

The Dashboard URL: https://192.168.91.235:8443, default username/password is admin/admin

Playbook run took 0 days, 0 hours, 51 minutes, 5 seconds
```

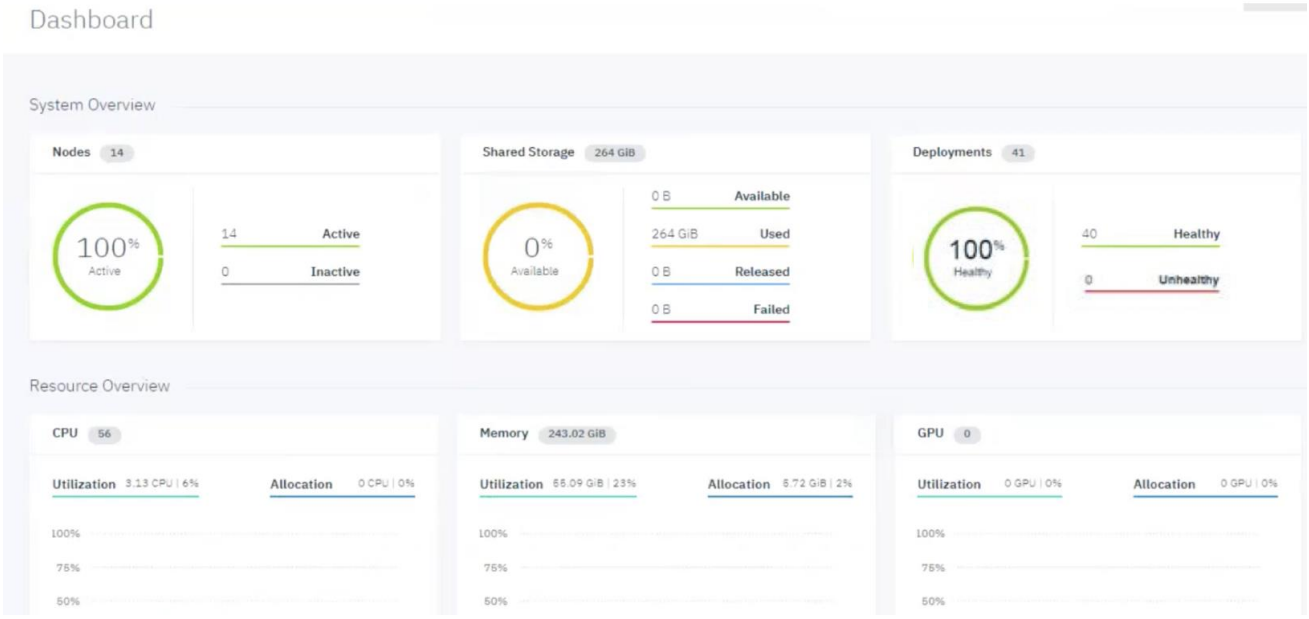
3. Point your browser to your cluster IP or hostname port 8443 to login to ICP. Use `admin` and `admin` for the user name and password which are the default values in `config.yaml` file.

Figure 58 ICP – Login Screen



4. The ICP Dashboard is loaded upon successful login.

Figure 59 ICP - Dashboard



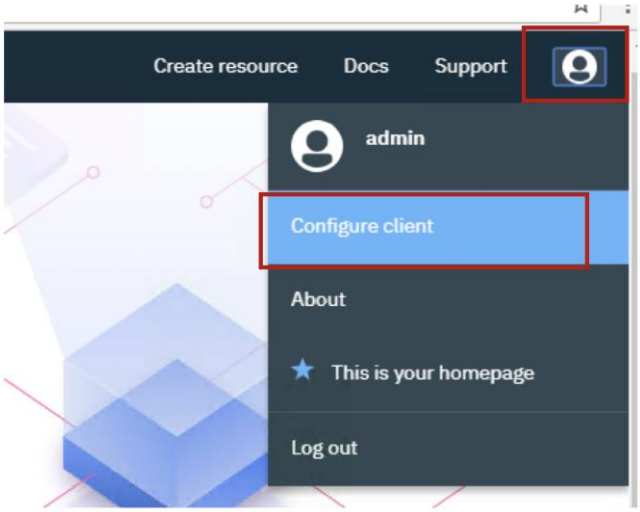
Accessing ICP Using kubectl

`Kubectl` is Kubernetes' CLI tool. To access the cluster via CLI, `kubectl` must be installed and configured. Version 1.9.1 of the `kubectl` is required as it matches the version of Kubernetes that is installed with ICP 2.1.0.2. The tool can be installed from the installation package of ICP, which includes the kubectl binary, or the binary can be downloaded from the Kubernetes site, see: https://v1-9.docs.kubernetes.io/docs/tasks/tools/install-kubectl/

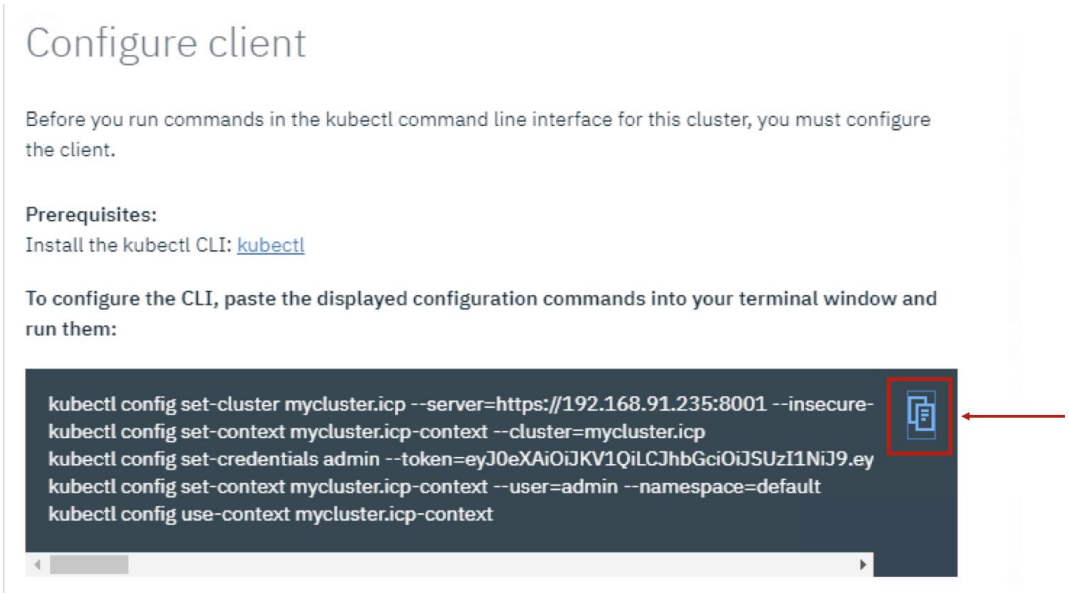
To use the binary from the ICP installer, complete the following steps:

1. From master1, run: docker run -e LICENSE=accept --net=host -v /usr/local/bin:/data/ibmcom/icp-inception:2.1.0.2-ee cp /usr/local/bin/kubectl /data.

2. Obtain the cluster configuration details from the management console of ICP. Log into your cluster and select User Name and Configure Client from the top right menu options.



3. Copy and paste the configuration information to your command line, and press Enter.



The `kubectl` configuration expires after 12 hours. You must log in and reconfigure `kubectl` every 12 hours. This limitation of the token expiration can be avoided by configuring a service account, see: https://www.ibm.com/developerworks/community/blogs/fe25b4ef-ea6a-4d86-a629-6f87ccf4649e/entry/Configuring_the_Kubernetes_CLI_by_using_service_account_tokens1?lang=en_us

NetApp Trident Installation and Configuration

This section provides a set of instructions about installing and configuring Trident in the Kubernetes cluster for ICP, and creating storage classes and provisioning PVCs. Trident version 18.04 is used in this reference, for the latest version and generic installation procedures of Trident please refer to: https://netapp-trident.readthedocs.io/en/stable-v18.04/kubernetes/deploying.html

As mentioned in the Solution Design section of this document, Trident is a dynamic storage provisioner for Kubernetes and it is completely integrated in the Kubernetes environment and run as a pod. Trident is used to dynamically provision and delete storage services within the Kubernetes' framework.

Proper configuration of the A300 backend storage system is required prior to the actual installation of Trident. These requirements are covered in the Solution Design section of this document.

Preparing the A300 Backend Storage

The AFF A300 storage system has to be designed and configured in a way that it can provide the required services for the ICP-Kubernetes cluster. These requirements will be defined in collaboration between the users of the ICP platform and the storage team. The storage design aspects are covered in the Solution Design section of this document.

Preparing the Nodes

As mentioned, both NFS and iSCSI storage options are available and supported by Trident and the AFF backend storage. Prior to installing Trident you need to enable NFS and iSCSI on the Ubuntu worker nodes. In our deployment procedure example, the focus is on NFS storage provisioning.

```
sudo apt-get install -y nfs-common
```

Installing iSCSI requires few steps; instructions can be found in the following link under iSCSI and Ubuntu / Debian:

https://netapp-trident.readthedocs.io/en/stable-v18.04/kubernetes/operations/tasks/worker.html#worker-preparation

Installing and Configuring Trident

Trident can be installed on any server with `kubectl` and access to the Kubernetes API.

In our solution example, we used the master1 node.

To install and configure Trident, complete the following steps:

1. From the server (master1 in our example), copy and extract the installation file, which is available from: https://github.com/NetApp/trident/releases/tag/v18.04.0

   https://netapp-trident.readthedocs.io/en/latest/

```
root@master1:~# ls -l
total 27400
drwxr-xr-x 9 root root     4096 Jul  5 07:25 icp-backup
-rw-r--r-- 1 root root        6 May 31 11:16 index.html
-rw-r--r-- 1 root root        6 May 31 11:16 index.html.1
drwxr-xr-x 2 root root     4096 Jul 10 11:04 netapp
-rw-r--r-- 1 root root      217 May 30 16:16 netconfig
drwxr-xr-x 5 1001 1001     4096 Jul 24 09:51 trident-installer
-rw-r--r-- 1 root root 28032627 Apr 19 18:23 trident-installer-18.04.0.tar.gz
```

2. Before running the installation script, we need configure temporary backend storage that the installer will use once to provision a volume to store its own metadata. Create a `backend.json` file in the setup directory. The file is shown below:

```
root@master1:~/trident-installer# cd setup/
root@master1:~/trident-installer/setup# ls -l
total 4
-rw-r--r-- 1 root root 235 Jun  8 10:11 backend.json
root@master1:~/trident-installer/setup# cat backend.json
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "192.168.50.203",
    "svm": "icp-svm",
    "username": "vsadmin",
    "password": "NetApp!23",
    "defaults": {
      "snapshotPolicy": "default"
    }
}
root@master1:~/trident-installer/setup#
```

This provides the backend storage information with the key attributes and values covered in the Solution Design.

3. Run the install command using the ***-n*** argument to specify the namespace for Trident to use, which in our case it is **trident**.

```
./tridentctl install -n trident
```

The installation should take about a minute and information about the various steps will be presented on the screen. If successful, the installation script will report "*Trident installation succeeded.*"

```
INFO Waiting for Trident REST interface.
INFO Trident REST interface is up.                    version=18.04.0
INFO Trident installation succeeded.
root@master1:~/trident-installer#
```

A dry-run option of the Trident installation is available and can be used to check for various issues prior to running the actual installation. The dry-run option can be executed as follows: `./tridentctl install -n trident --dry-run -d`

```
root@master1:~/trident-installer# ./tridentctl install -n trident --dry-run -d
INFO Storage driver loaded.                           driver=ontap-nas
INFO Dry run completed, no problems found.
root@master1:~/trident-installer# ./tridentctl install -n trident
```

4. With the successful installation you can obtain the details of the trident pod `(kubectl get pods -n trident)` and as shown in 86, the Trident pod is running and it has 2/2 containers (which are the trident-main and its etcd).

Figure 60 Trident Pods

```
root@master1:~# kubectl get pods -n trident
NAME                       READY     STATUS    RESTARTS   AGE
trident-7476f4d644-6bt8z   2/2       Running   0          42s
root@master1:~#
```

5. The access information to the backend A300 storage system from Kubernetes is defined in .json files. The storage classes for Kubernetes to use are defined in the .yaml files. Sample files are provided with the Trident installation in a directory named `sample-input`. You will need to create or modify these files.

Figure 61 Sample Definition json and yaml files for Trident

```
root@master1:~/trident-installer/sample-input# ls -l
total 80
-rw-r--r-- 1 1001 1001 276 Apr 19 18:19 backend-eseries-iscsi.json
-rw-r--r-- 1 1001 1001 214 Apr 19 18:19 backend-ontap-nas-economy.json
-rw-r--r-- 1 1001 1001 246 Apr 19 18:19 backend-ontap-nas.json
-rw-r--r-- 1 1001 1001 271 Apr 19 18:19 backend-ontap-san-full.json
-rw-r--r-- 1 1001 1001 206 Apr 19 18:19 backend-ontap-san.json
-rw-r--r-- 1 1001 1001 540 Apr 19 18:19 backend-solidfire.json
-rw-r--r-- 1 1001 1001 220 Apr 19 18:19 pvc-basic-beta.yaml
-rw-r--r-- 1 1001 1001 242 Apr 19 18:19 pvc-basic-clone.yaml
-rw-r--r-- 1 1001 1001 180 Apr 19 18:19 pvc-basic.yaml
-rw-r--r-- 1 1001 1001 162 Apr 19 18:19 pvc-default-class.yaml
-rw-r--r-- 1 1001 1001 512 Apr 19 18:19 pvc-full.yaml
-rw-r--r-- 1 1001 1001 155 Apr 19 18:19 storage-class-basic-v1beta1.yaml.templ
-rw-r--r-- 1 1001 1001 150 Apr 19 18:19 storage-class-basic.yaml.templ
-rw-r--r-- 1 1001 1001 233 Apr 19 18:19 storage-class-bronze-default.yaml
-rw-r--r-- 1 1001 1001 211 Apr 19 18:19 storage-class-ontapnas-gold.yaml
-rw-r--r-- 1 1001 1001 196 Apr 19 18:19 storage-class-ontapnas-k8s1.8-mountoptions.yaml
-rw-r--r-- 1 1001 1001 175 Apr 19 18:19 storage-class-silver.json
-rw-r--r-- 1 1001 1001 191 Apr 19 18:19 storage-class-solidfire-bronze.yaml
-rw-r--r-- 1 1001 1001 224 Apr 19 18:19 volume-full.json
-rw-r--r-- 1 1001 1001 118 Apr 19 18:19 volume.json
root@master1:~/trident-installer/sample-input#
```

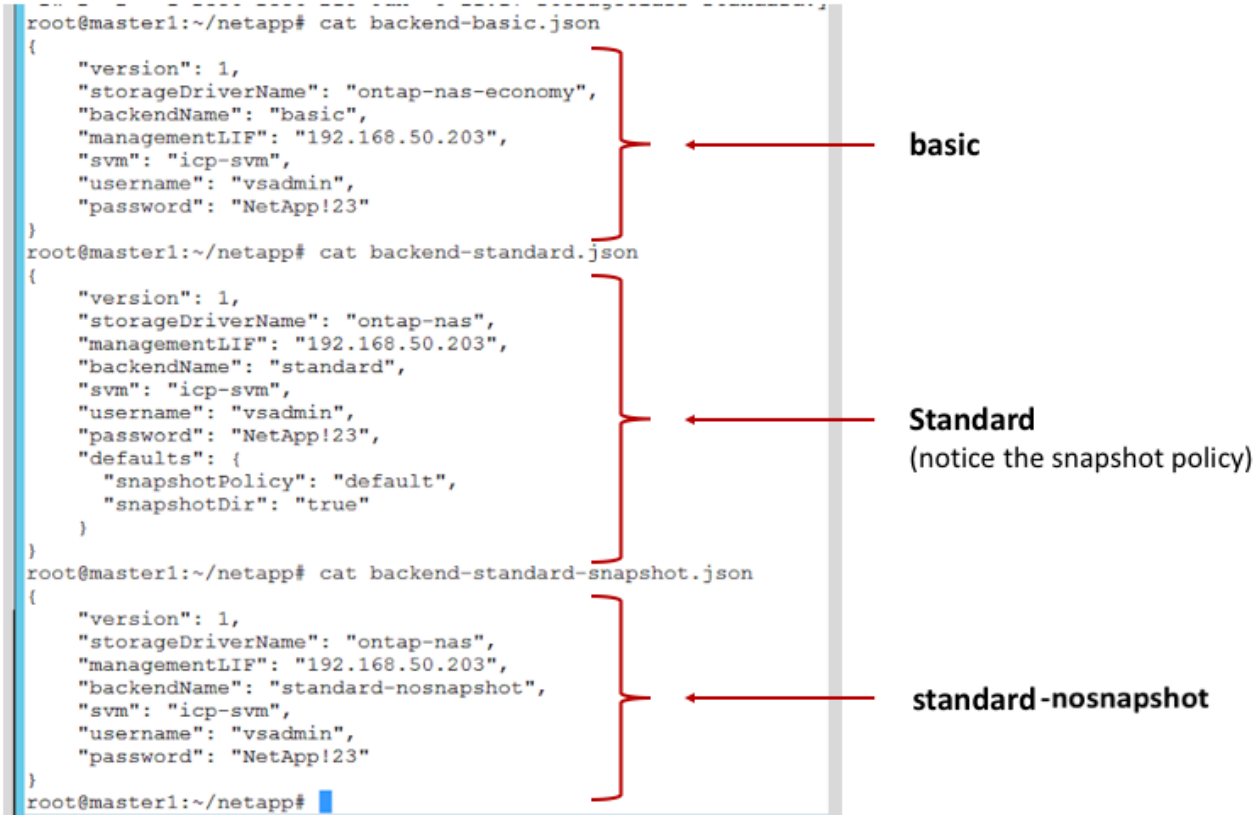
In our solution, you defined six files for the storage services. Three are .json files with the backend storage information, and three .yaml files with the corresponding storage classes.

Figure 62 Storage Definition Files for Trident

```
root@master1:~/netapp# ls -l
total 28
-rw-r--r-- 1 root root 212 Jun  8 11:07 backend-basic.json
-rw-r--r-- 1 root root 295 Jun  8 11:15 backend-standard.json
-rw-r--r-- 1 root root 218 Jul 10 11:02 backend-standard-snapshot.json
-rw-r--r-- 1 root root 510 Jun 26 10:32 IPs
-rw-r--r-- 1 root root 217 Jun  8 11:09 storageclass-basic.yaml
-rw-r--r-- 1 root root 233 Jul 10 11:04 storageclass-standard-nosnap.yaml
-rw-r--r-- 1 root root 215 Jun  8 11:17 storageclass-standard.yaml
root@master1:~/netapp# cat storageclass-basic.yaml
```

6. We created the following three .json files: `backend-basic.json`, `backend-standard.json` and `backend-standard-snapshot.json`. The files were created in a directory named `netapp`**.** Let us examine the content of each file as the examples used in our test environment.

Figure 63 Storage Backend Definition Files



7. We also created the following three .yaml files: `storageclass-basic.yaml`, `storageclass-standard.yaml` and `storageclass-standard-nosnap.yaml` in the netapp directory.

Figure 64 Storage Classes

```
root@master1:~/netapp# cat storageclass-basic.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: basic
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas-economy"
  storagePools: "basic:aggr1_flexpod_icp_01,aggr1_flexpod_icp_02"
```

Storageclass-basic

```
root@master1:~/netapp# cat storageclass-standard.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas"
  storagePools: "standard:aggr1_flexpod_icp_01,aggr1_flexpod_icp_02"
```

Storageclass-standard

```
root@master1:~/netapp# cat storageclass-standard-nosnap.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard-nosnap
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas"
  storagePools: "standard-nosnapshot:aggr1_flexpod_icp_01,aggr1_flexpod_icp_02"
```

Storageclass-standard-nosnap

8. You can list the available backend storage services by using:

```
tridentctl -n trident get backend t
```

```
root@master1:~/netapp# ../trident-installer/tridentctl -n trident get backend
+----------------------+--------------------+--------+---------+
|         NAME         |  STORAGE DRIVER    | ONLINE | VOLUMES |
+----------------------+--------------------+--------+---------+
| basic                | ontap-nas-economy  | true   |       1 |
| standard             | ontap-nas          | true   |       5 |
| standard-nosnapshot  | ontap-nas          | true   |       0 |
+----------------------+--------------------+--------+---------+
root@master1:~/netapp#
```

9. Trident is now installed, configured with our backend storage options and we can start using it with our three storage classes within Kubernetes. Please refer to the Validation section of the document to learn how Trident was tested within ICP.

Applications Deployment on FlexPod for Private Cloud

A number of containerized applications can be deployed that are available in the ICP catalog; in our solution, we deployed WebSphere Liberty and Jenkins applications on ICP platform using the HELM charts.

Deploy Application from Helm Catalog

This section reviews the steps to deploy containerized applications using Helm Catalog.

Deploying WebSphere Liberty from Helm Charts with Dynamically Provisioned Storage (via Trident)

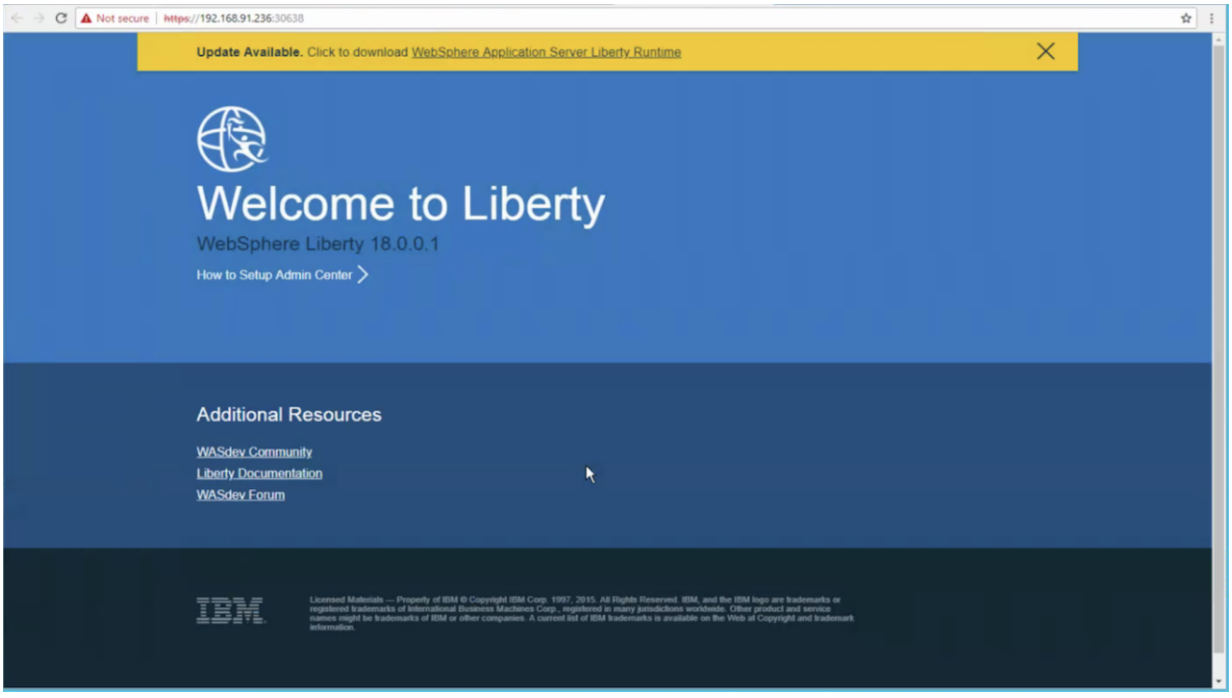
For the purpose of this validation, we used WebSphere Liberty, which is a containerized version of WebSphere Network Deployment. Liberty is included as an out-of-the-box application with ICP.

Trident is required to be installed before you can start deploying Liberty. Also, in this solution, we have already defined and configured storage classes to be used for the persistent storage.

1. From the IBM Cloud Private console, click Menu > Catalog.
2. Click Configure.
3. Provide a proper release name in the Release name field and select the I have read and agreed to the license agreements check box.
4. Specify useDynamicProvisioning as **true**.
5. Continue in the same configuration web UI and in the "**Storage Class Name**" section provide the storage class name exactly as defined in one of the appropriate storageClass YAML files. For more information, refer to Figure 62 through Figure 64.

Provide the appropriate storage class based on the application needs.

6. In the "Resource Configuration" section of the same web UI, change Memory and CPU if required optionally.
7. Click Install.
8. Validate the deployment by clicking Workloads >Helm releases in IBM Cloud Private.
9. Verify Liberty is running and accessible.

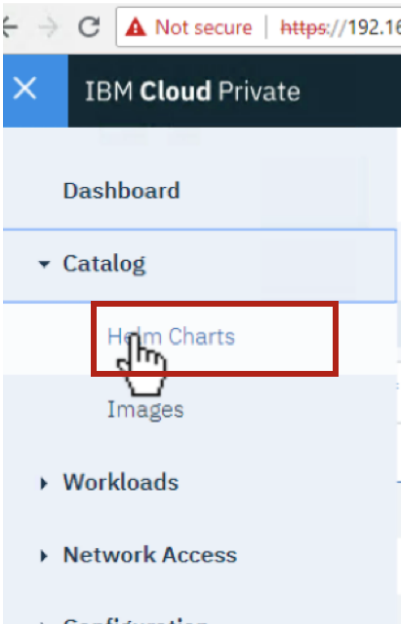


Deploy Jenkins from Helm Charts with Dynamically Provisioned Storage (via Trident)

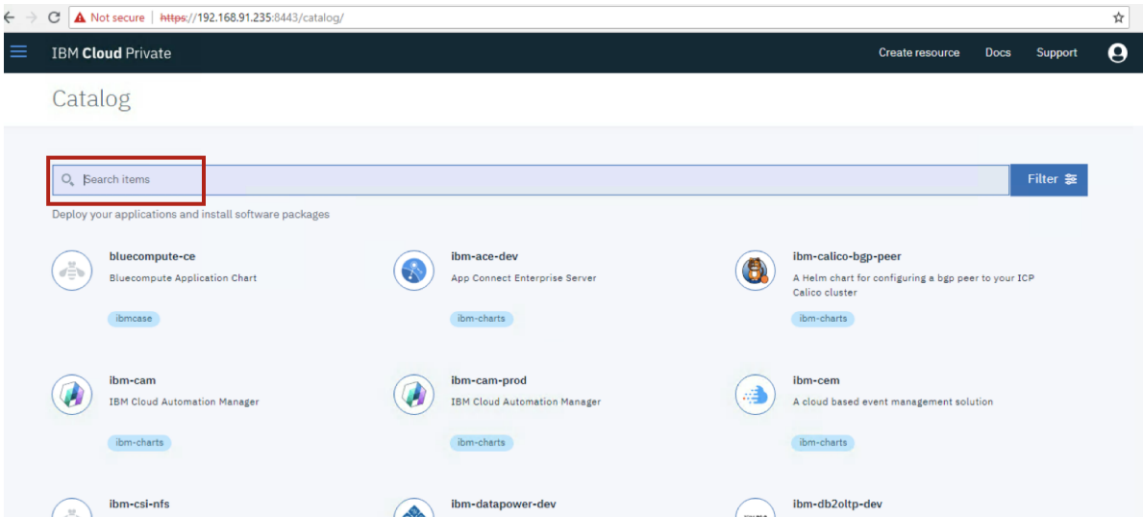
Jenkins is a leading open source platform for building, deploying and automating development projects. We tested a comprehensive deployment with Jenkins.

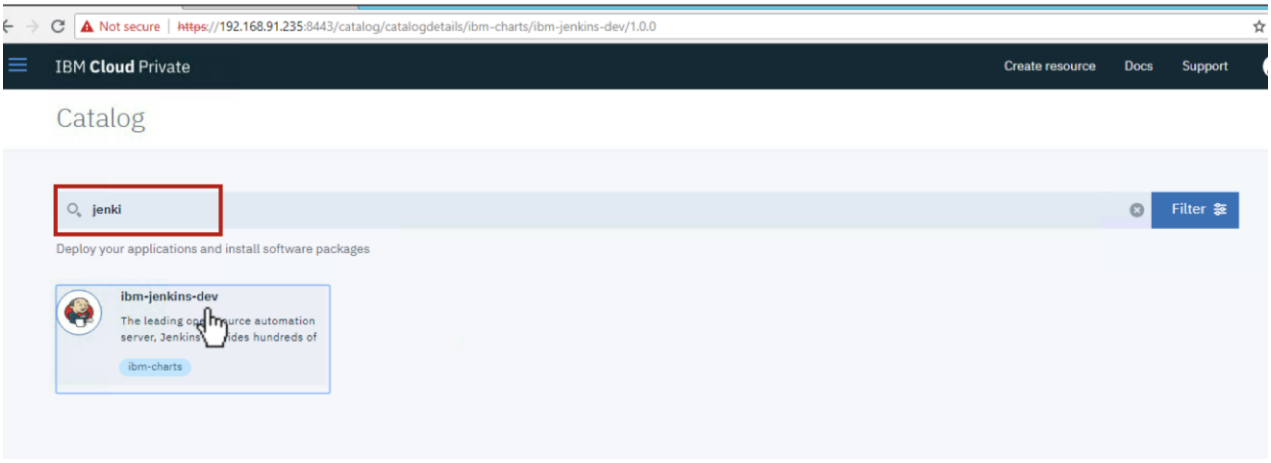
To deploy Jenkins from the Helm Charts, complete the following steps:

1. From the IBM Cloud Private menu, click **Catalog** and then click **Helm Charts**.



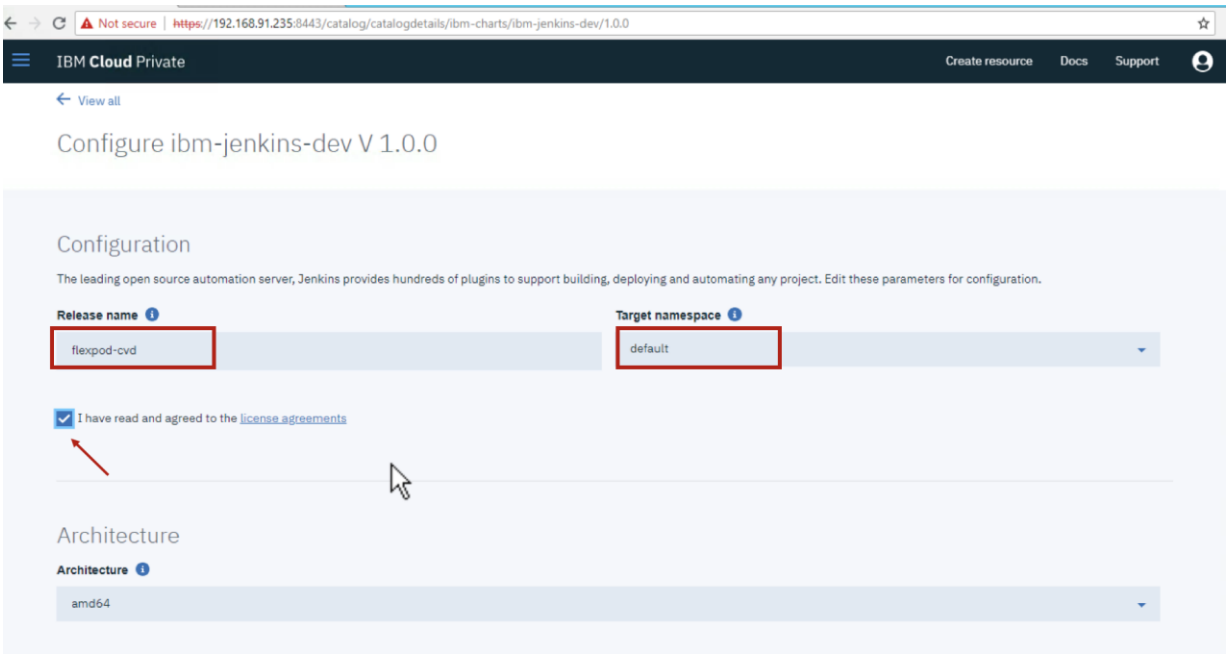
2. You will see many charts already populated with ICP, select Jenkins.





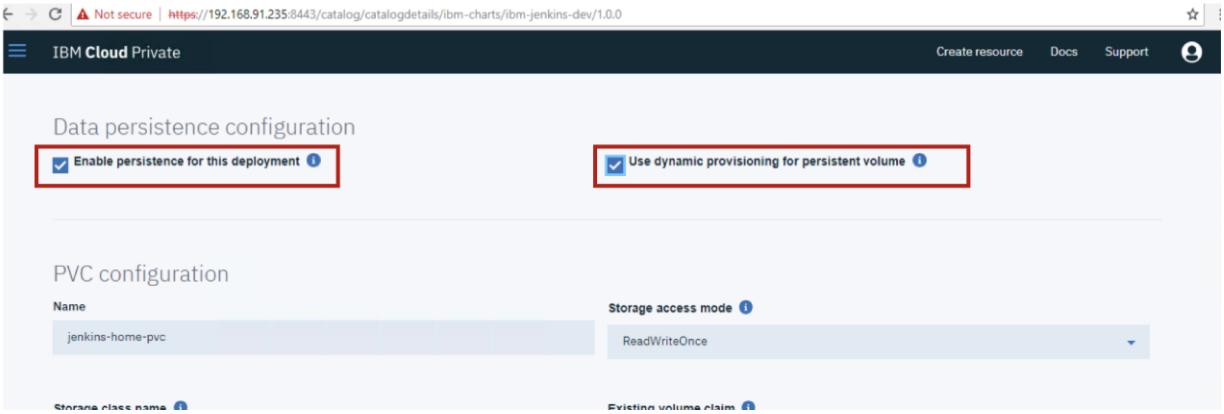
3. At the Jenkins configuration page enter the following values:

   a. The Release name: **flexpod-cvd** and the name space we used is **default.** Make sure to check the “I have read and agreed to the license agreement.”

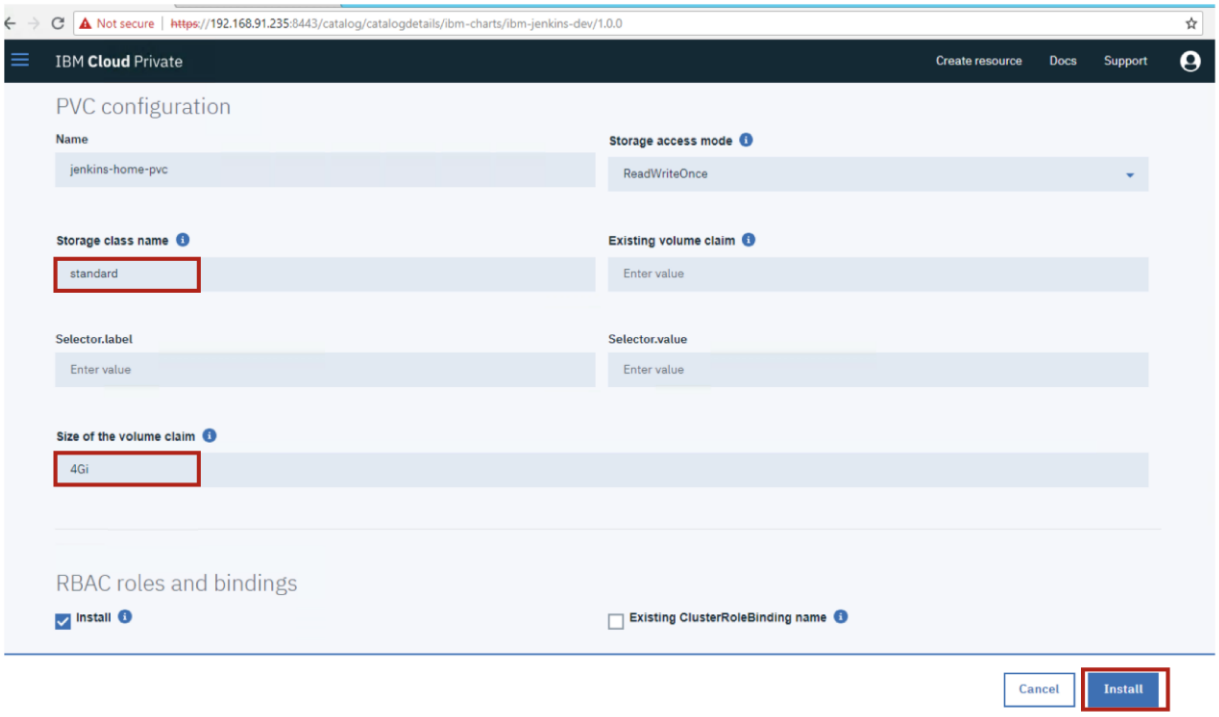


4. Scroll down to enter additional value about the persistent storage. Make sure the boxes are checked for the persistent storage options.

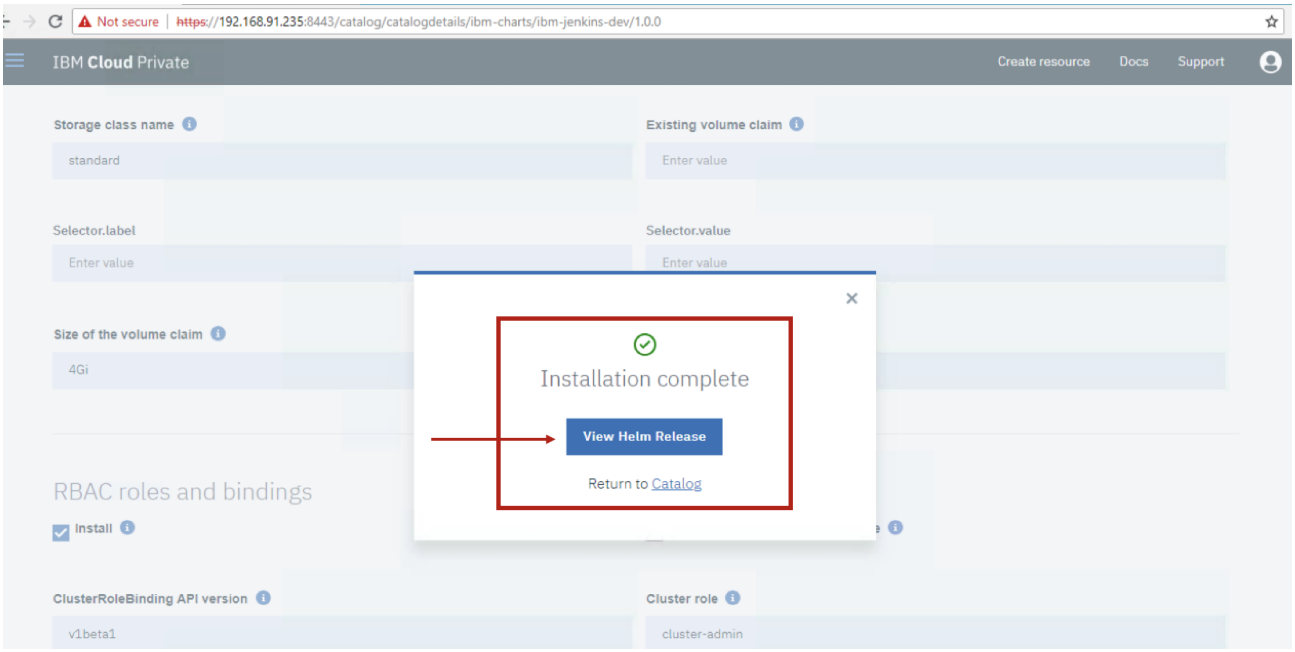
Figure 65 Persistent Storage Options



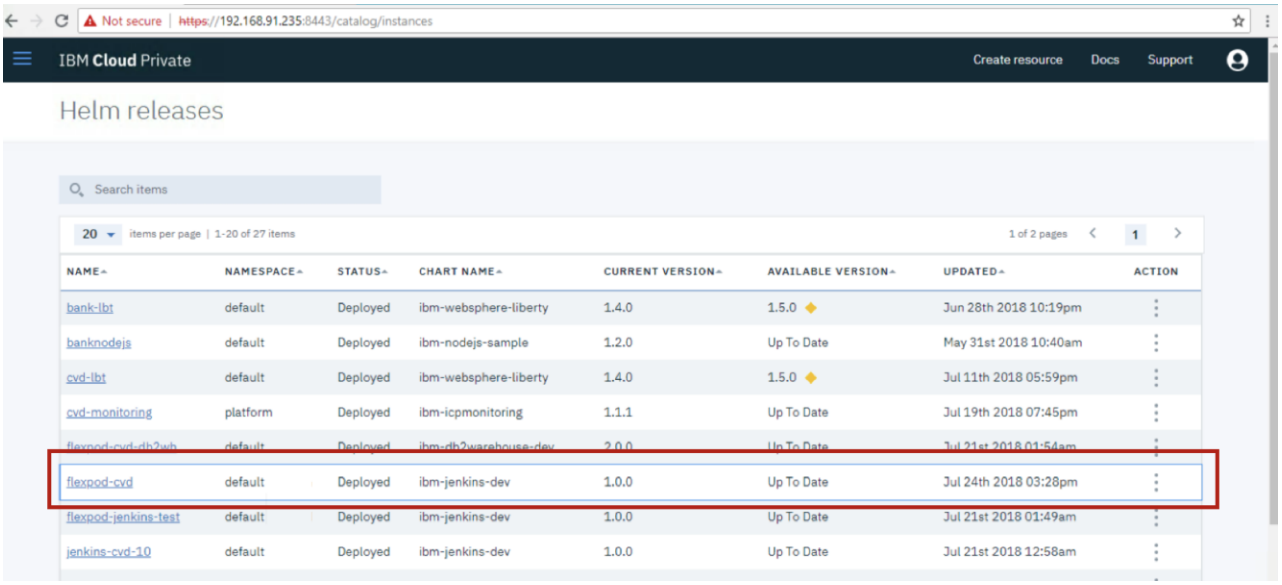
5. Continue to enter additional values for the storage class; enter the value: **standard** (see the Solution Design section for Trident) and PCV size as **4** Gi. Click Install.



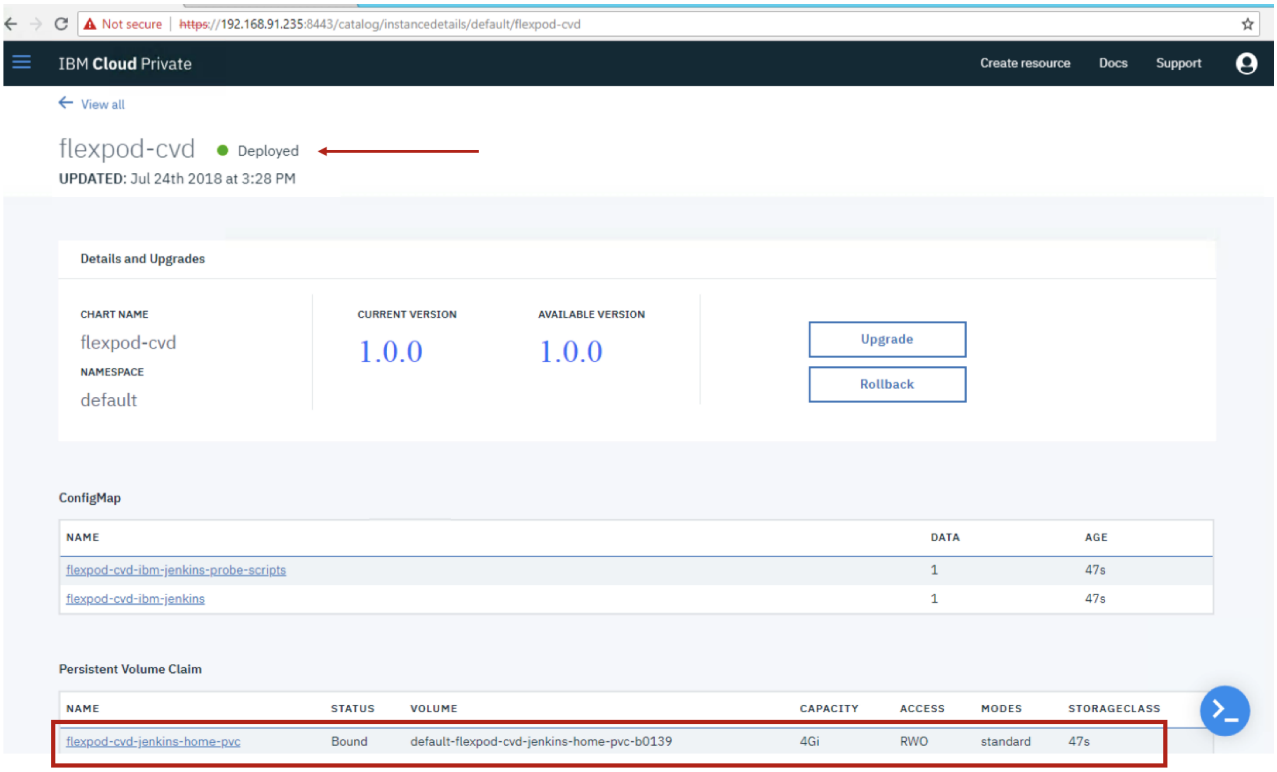
6. The installation is complete. Go to the **Helm Release** to check the status.



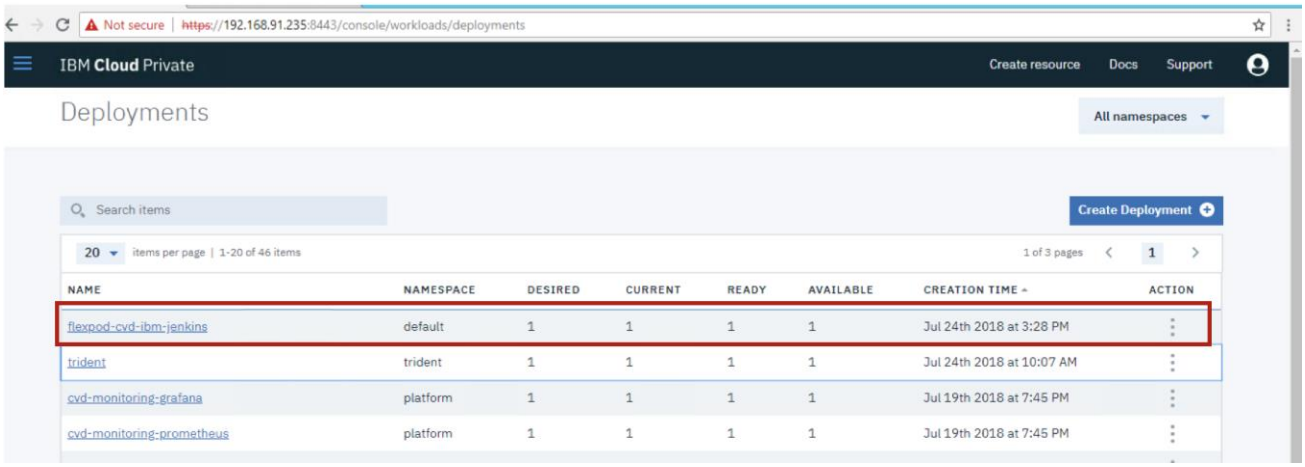
7. The deployment is listed in the Helm Releases.



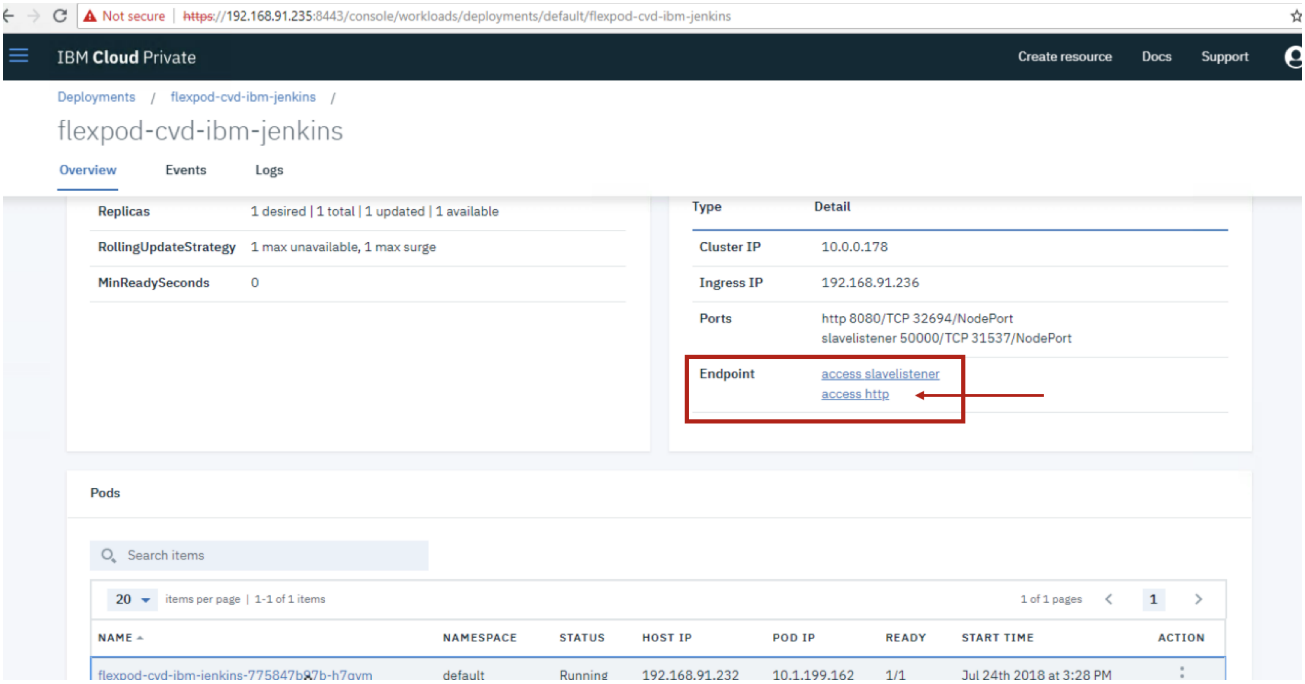
8. View the details. You can see the status is Deployed and the Persistent Volume Claim.



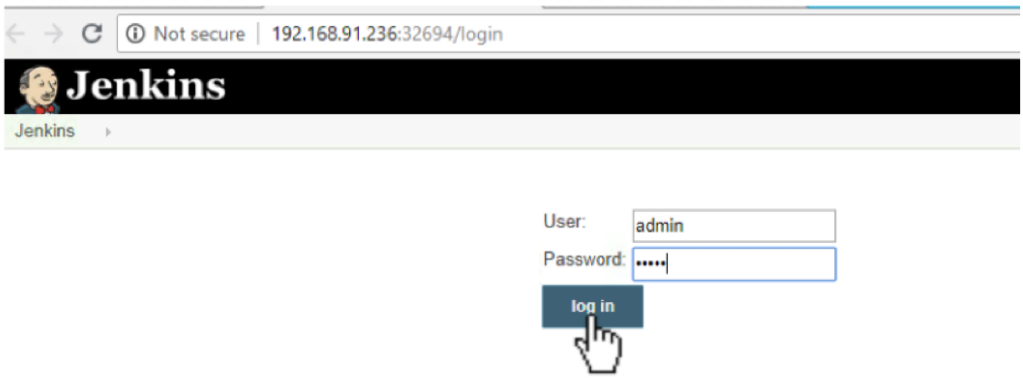
9. From the **Workload** menu option select **Deployments** and click the Jenkins' deployment line item.



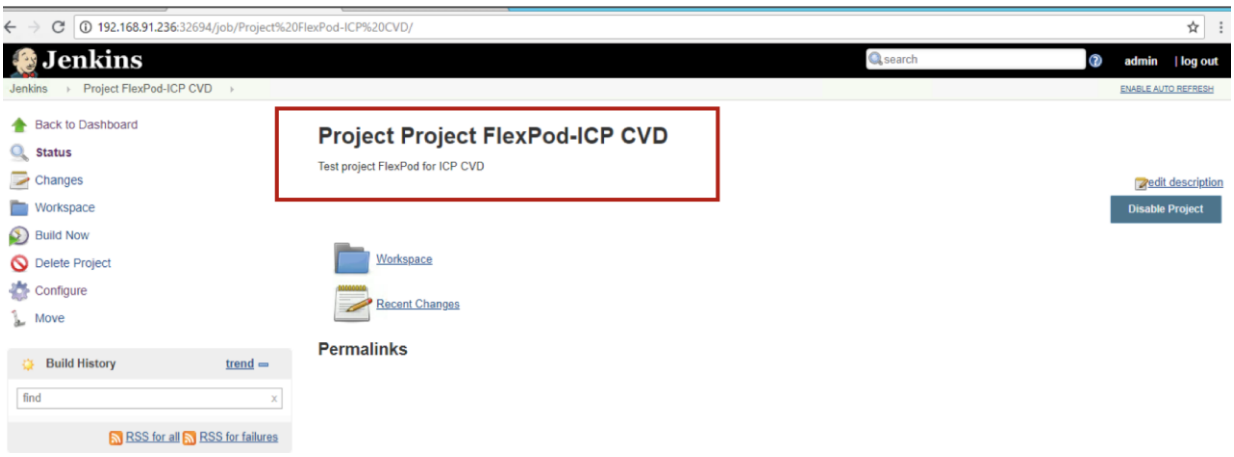
10. The **Endpoint** to connect to is displayed. Click the **access http** to access Jenkins.



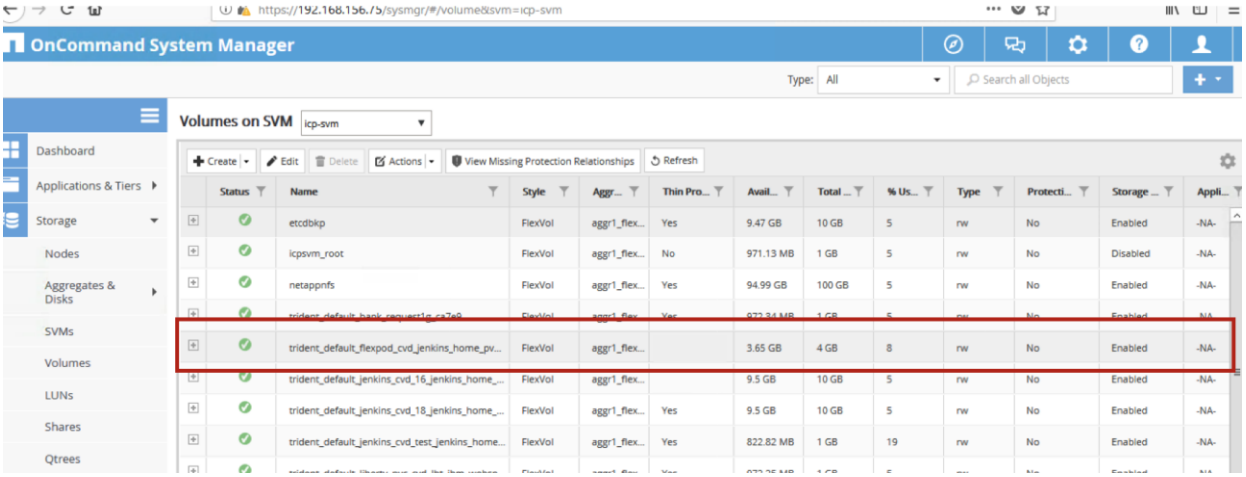
11. Use admin/admin to log in to Jenkins.



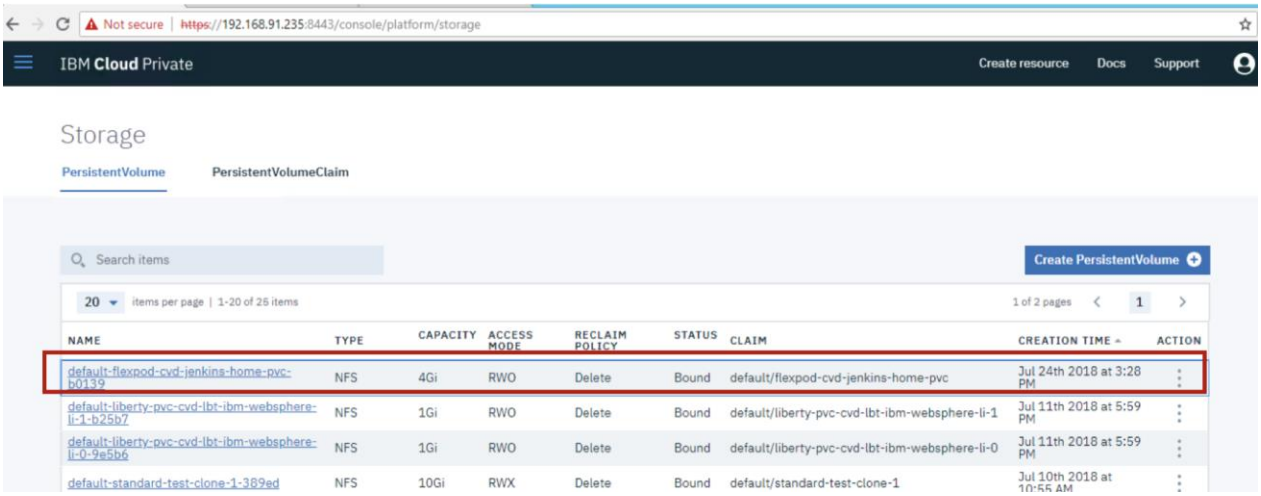
We created a project which we used for other testing purposes as shown below:



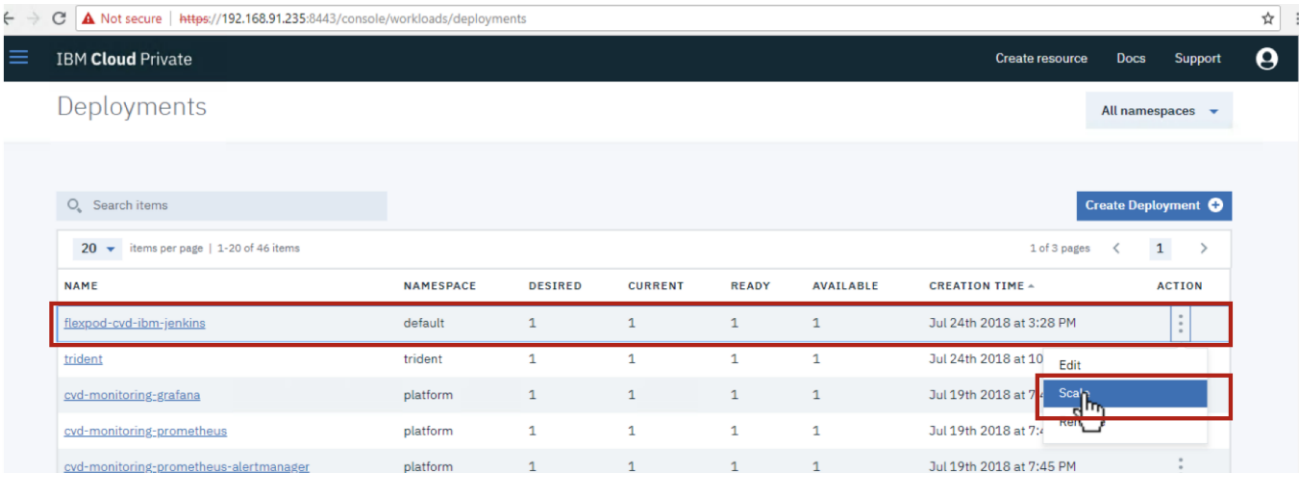
12. At the storage system, check the status in OnCommand System Manager. You can see the volume created by Trident. Trident generated volumes have the prefix of: “trident”, followed by the namespace used (**default** in our case), then the name of the release and the name of the pvc.

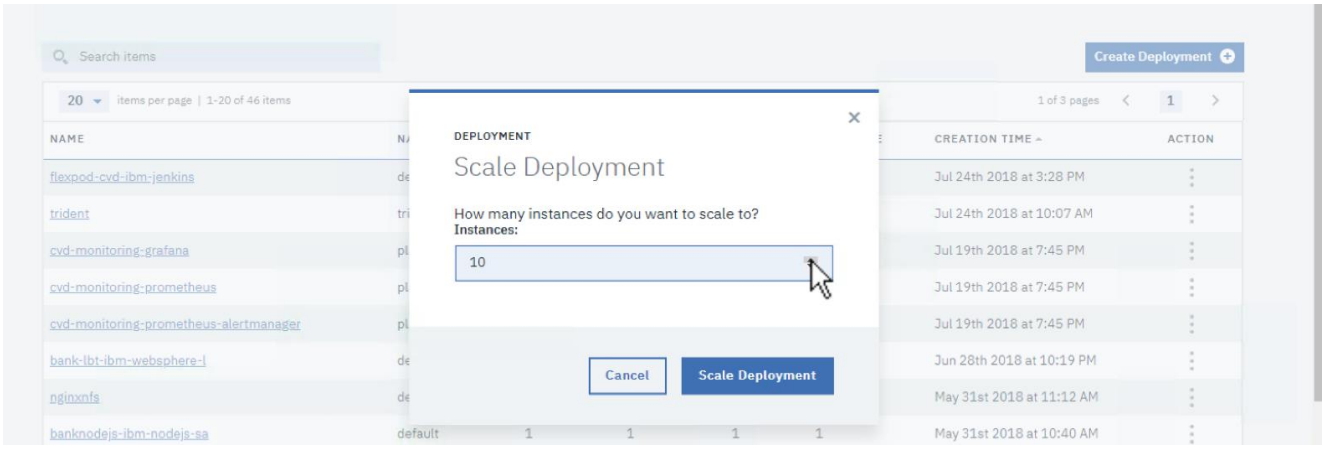


13. View the PV in the storage menu option in ICP.

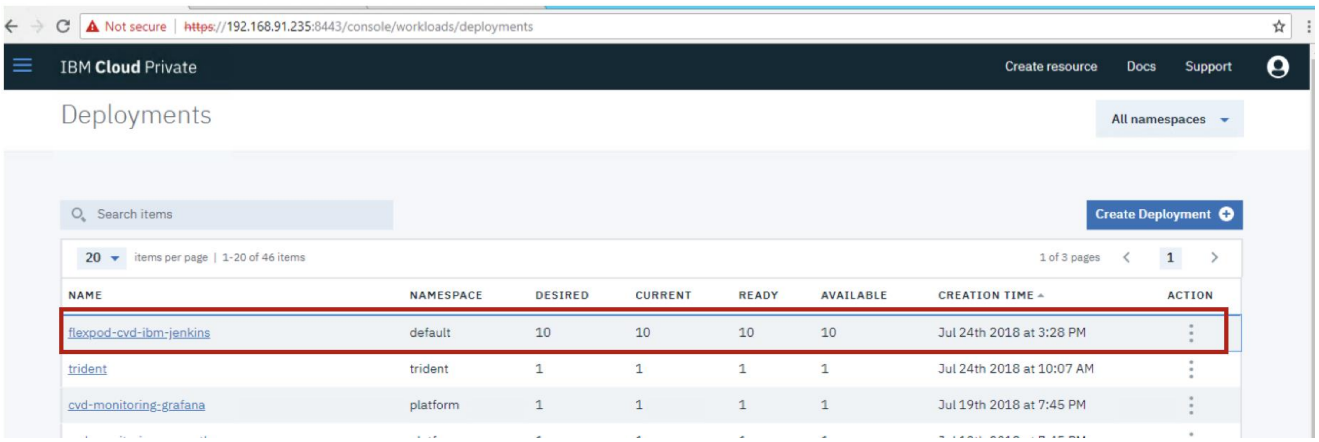


14. You can scale the deployment from 1 instance to 10.

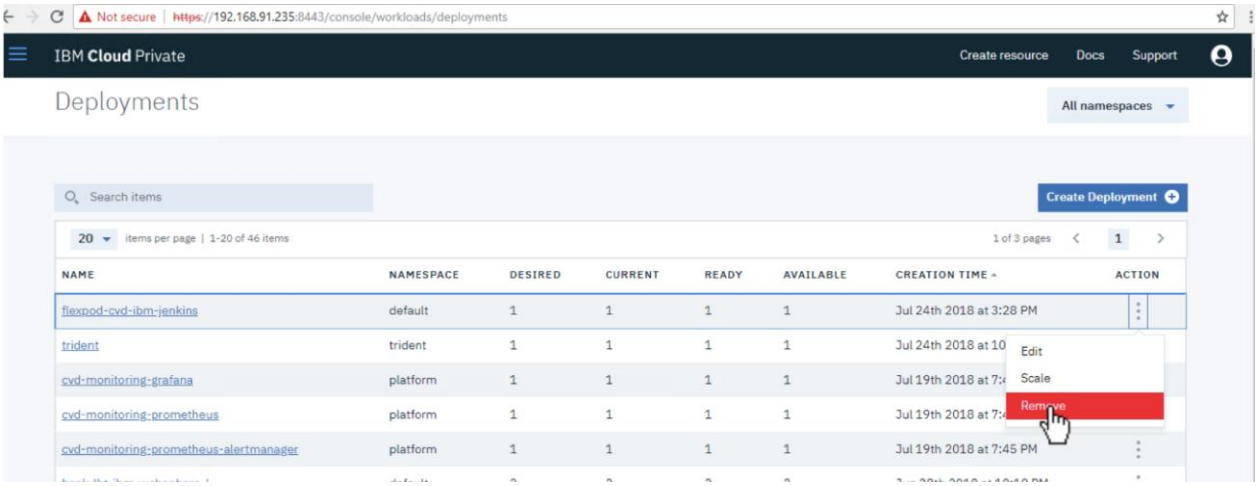




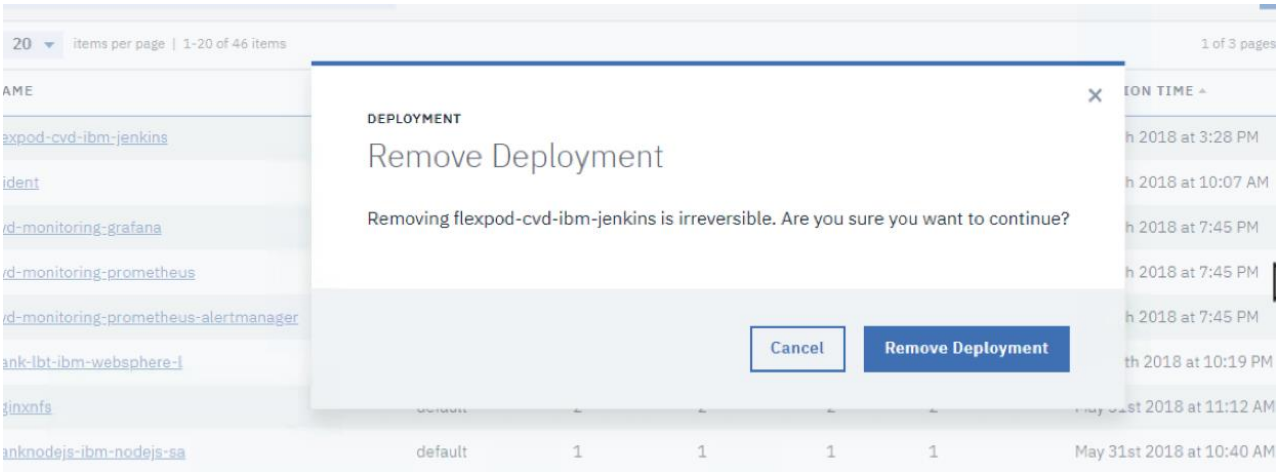
15. There are now 10 instances. Scale down to 1 and then continue to remove the Jenkins deployment completely.



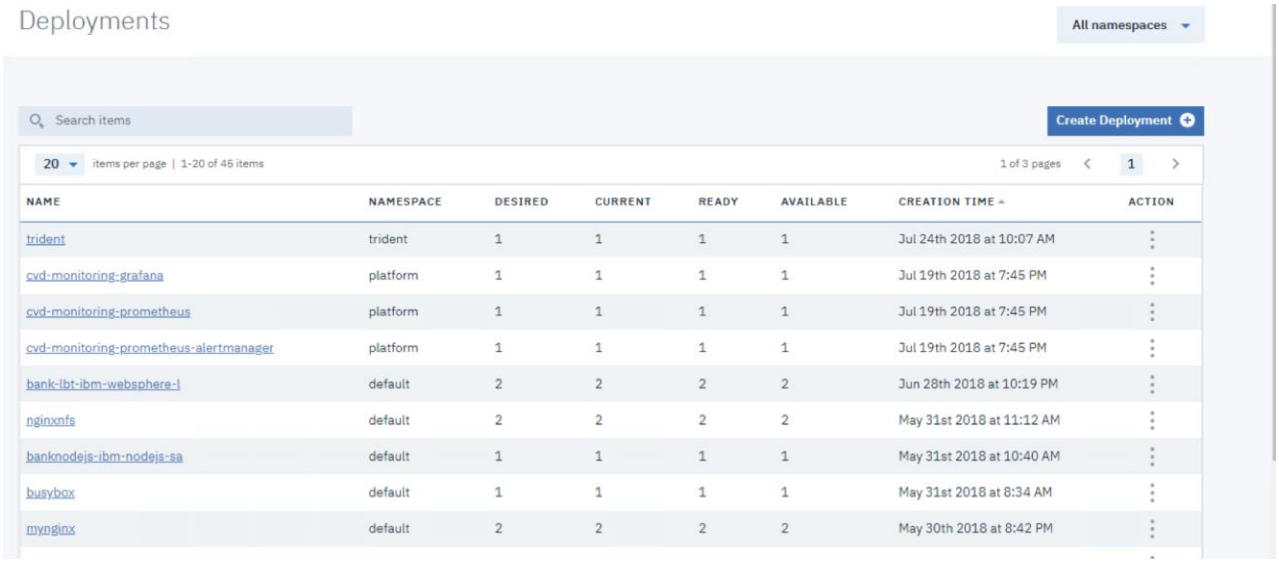
16. Remove Jenkins by selecting Remove from the Action drop-down list.



17. Confirm the removal of the deployment.



The Jenkins deployment was removed as shown below:



18. Refresh the list of volumes in the System Manager and see that the Jenkins volume has also been removed.

Figure 66 Jenkins' Volume Still Shown Prior to Refresh

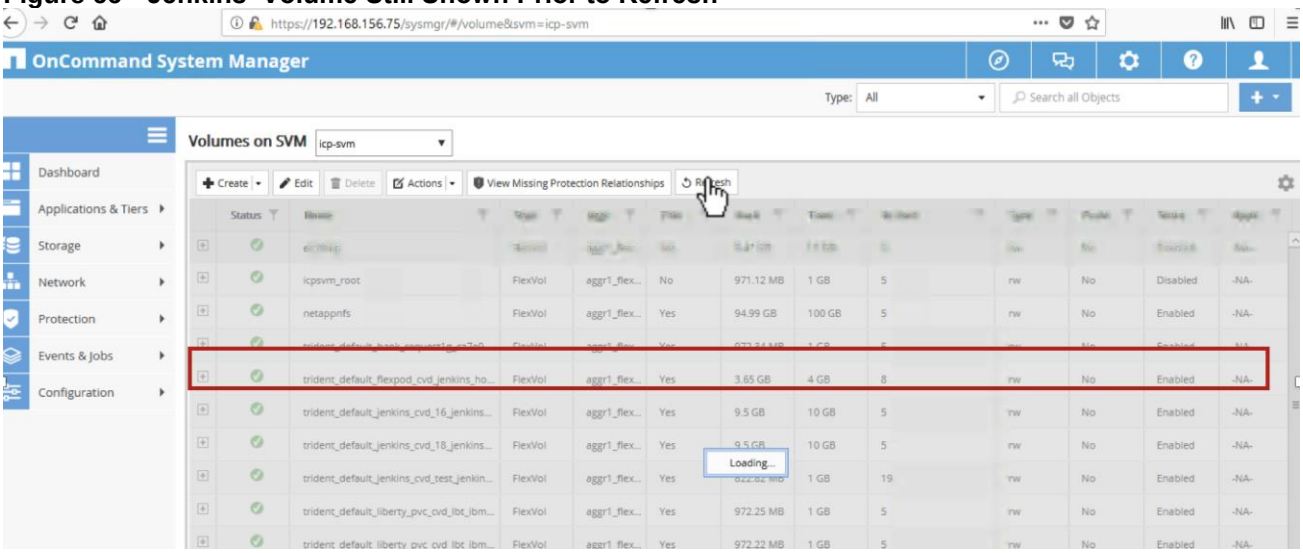
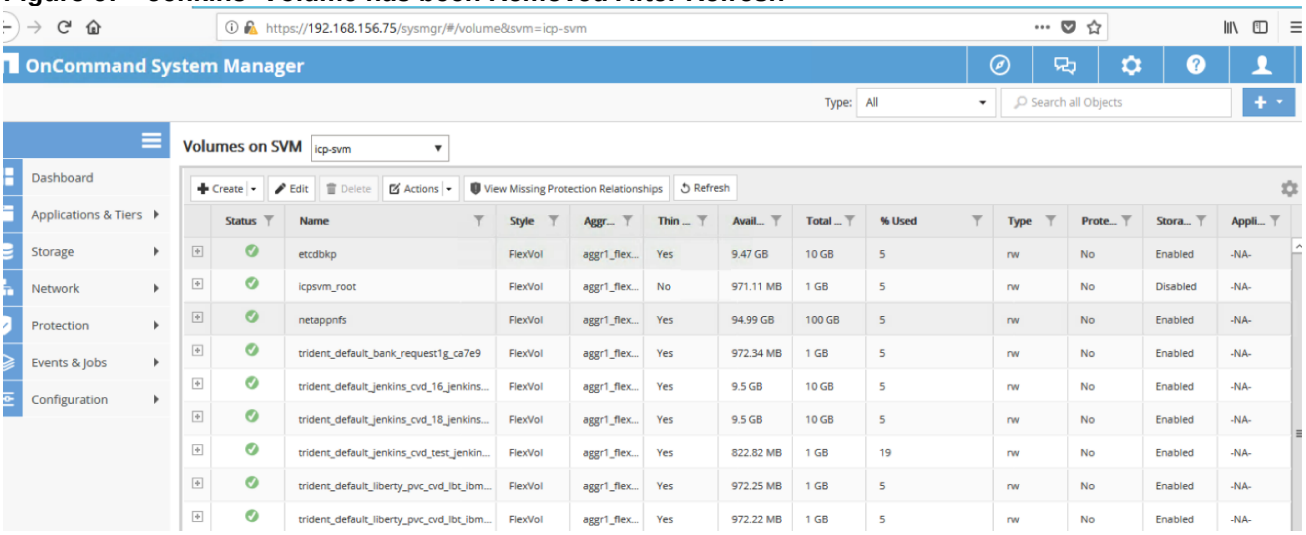


Figure 67 Jenkins' Volume has been Removed After Refresh

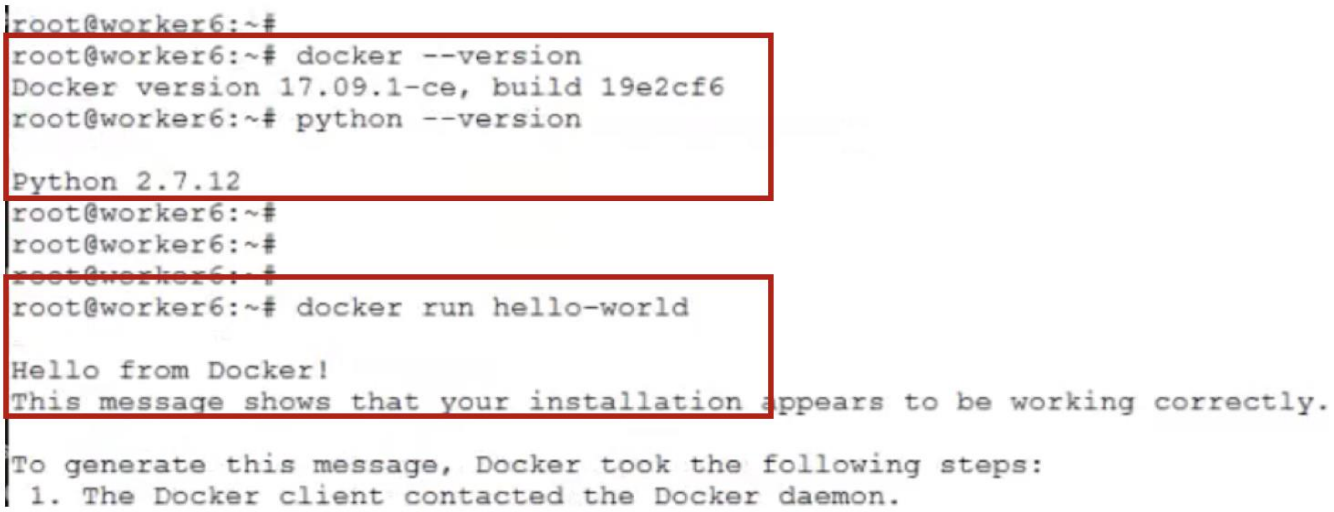


IBM Cloud Private Day-2 Operations

Add New Worker Node

Adding a Worker node is required to support additional workloads as the environment grows. When determined that additional worker node is needed, a new VM has to be deployed and configured, and then added to the Kubernetes cluster as a resource. Deploying and configuring the VM is covered in the earlier in the steps of preparing and installing ICP. In addition, the etc/hosts file on all other nodes has to be updated with the information of the new Worker node (192.168.91.238 in our case).

Figure 68 Validating Docker and Python New Worker Node



When the new VM is ready, you can run the installation script from the master1, the node that has the config.yaml file. The command to run is:

```
docker run -e LICENSE=accept --net=host \
-v "$(pwd)":/installer/cluster \
ibmcom/icp-inception:2.1.0.2-ee worker -l \
192.168.91.238
```

Figure 69 Adding Node Script

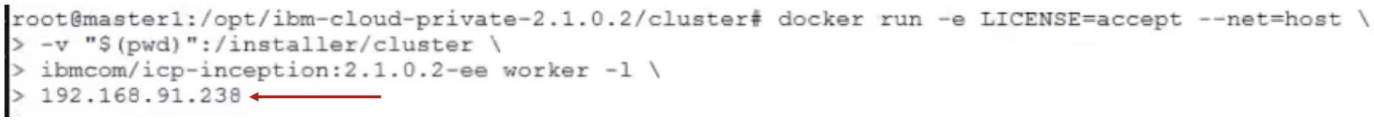


Figure 70 Deployment of New Worker Node In Progress

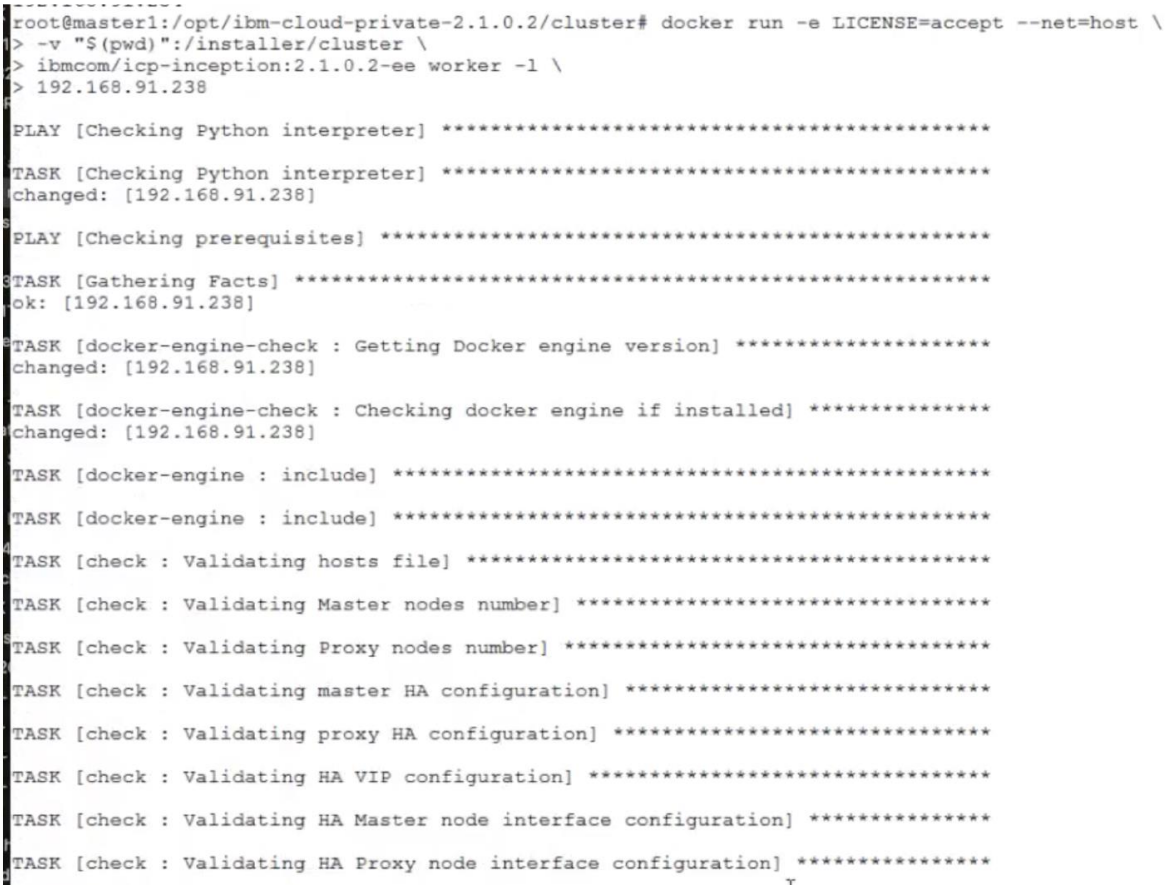
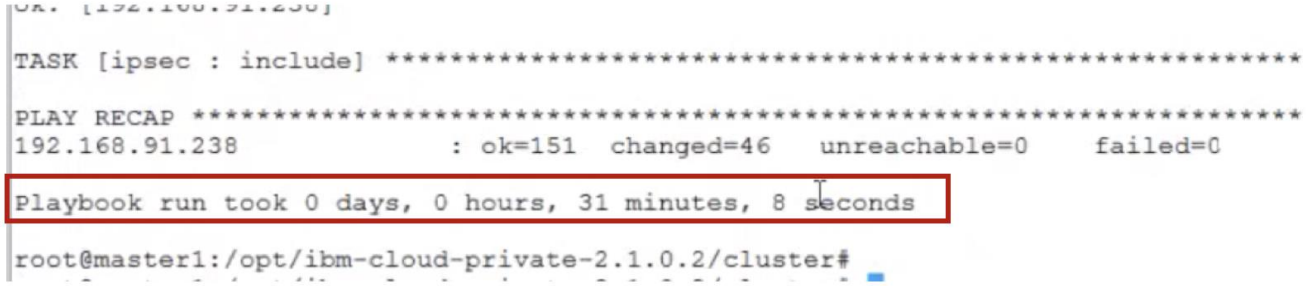
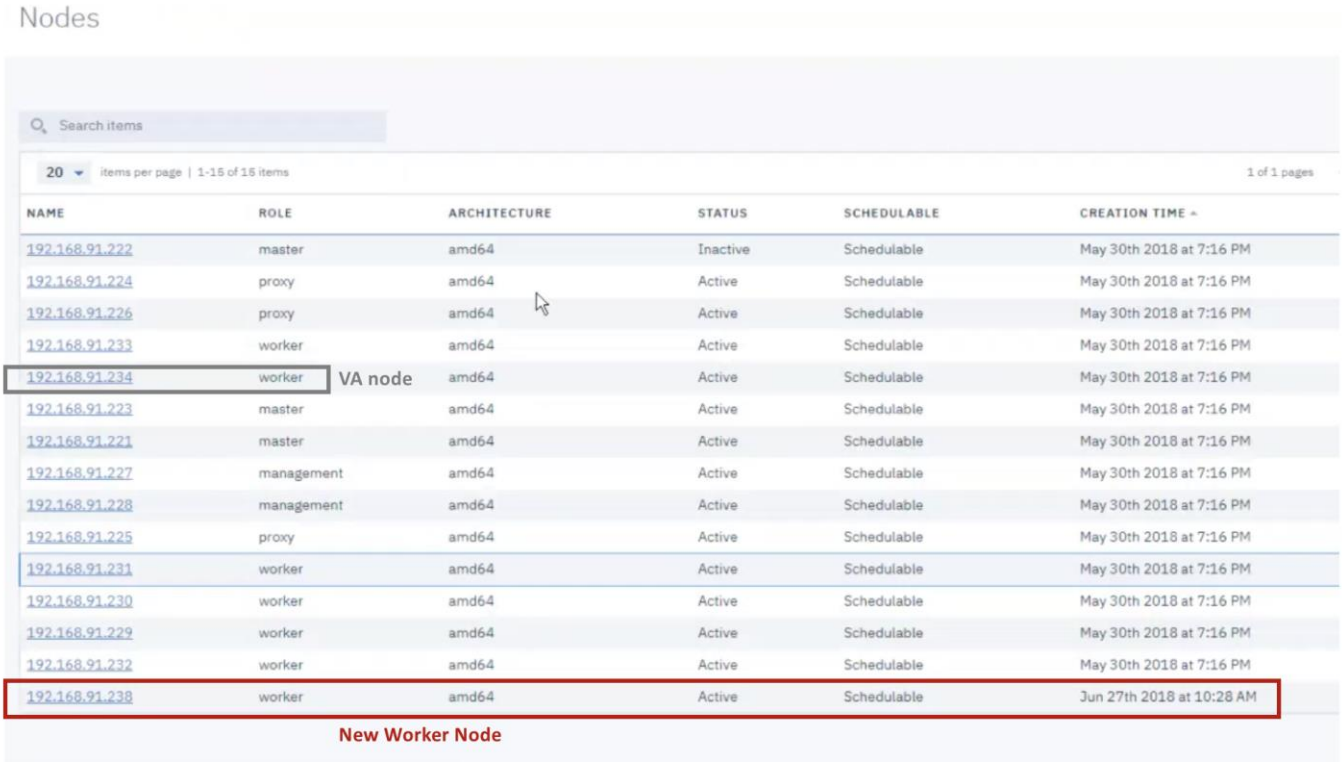


Figure 71 Installation Completed



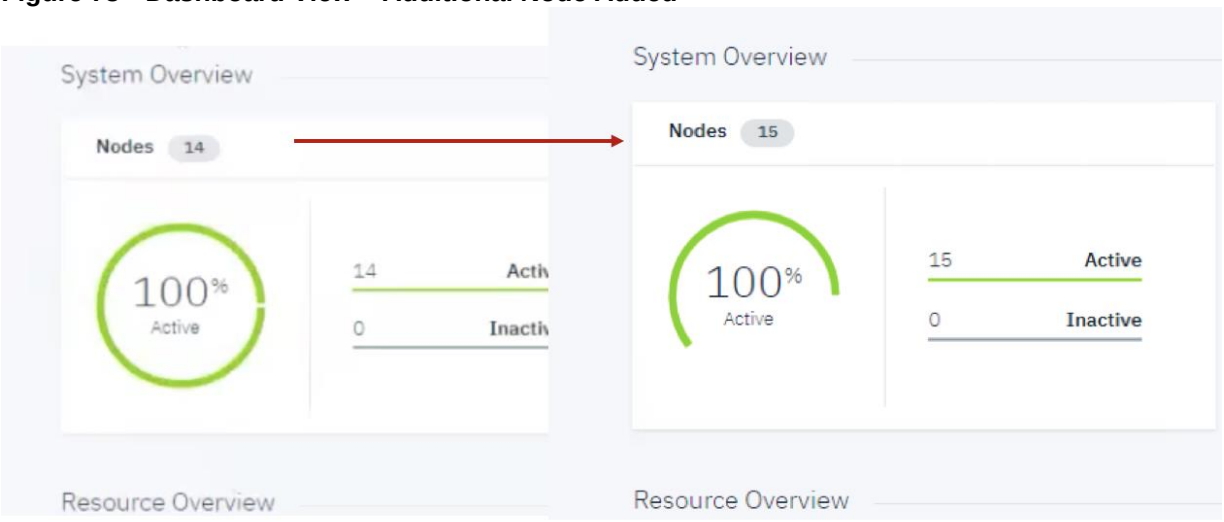
You now have the sixth Worker node added to ICP.

Figure 72 New Worker Node Listed in ICP



Total count of nodes is now 15, up by one.

Figure 73 Dashboard View – Additional Node Added



Remove Worker Node

Removing a Worker node can be done by running the uninstall script from master1:

```
docker run -e LICENSE=accept --net=host \
-v "$(pwd)":/installer/cluster \
ibmcom/icp-inception:2.1.0.1-ee uninstall -l \
```

```
ip_address_of _Worker_Node
```

Validation

Test Plan

This section provides the details about the tests conducted by the team, validating the design, and the implementation aspects of this solution.

A high-level summary of the FlexPod Datacenter with IBM Cloud Private validation is provided in this section.

FlexPod Infrastructure Validation

The system was validated for resiliency by failing various aspects of the system under load. Examples of the types of tests executed include:

- Failure and recovery of iSCSI booted ESXi hosts in a cluster
- Rebooting of iSCSI booted hosts
- Service Profile migration between blades
- Failure of partial and complete IOM links
- Failure and recovery of iSCSI paths to AFF nodes, Nexus switches, and fabric interconnects

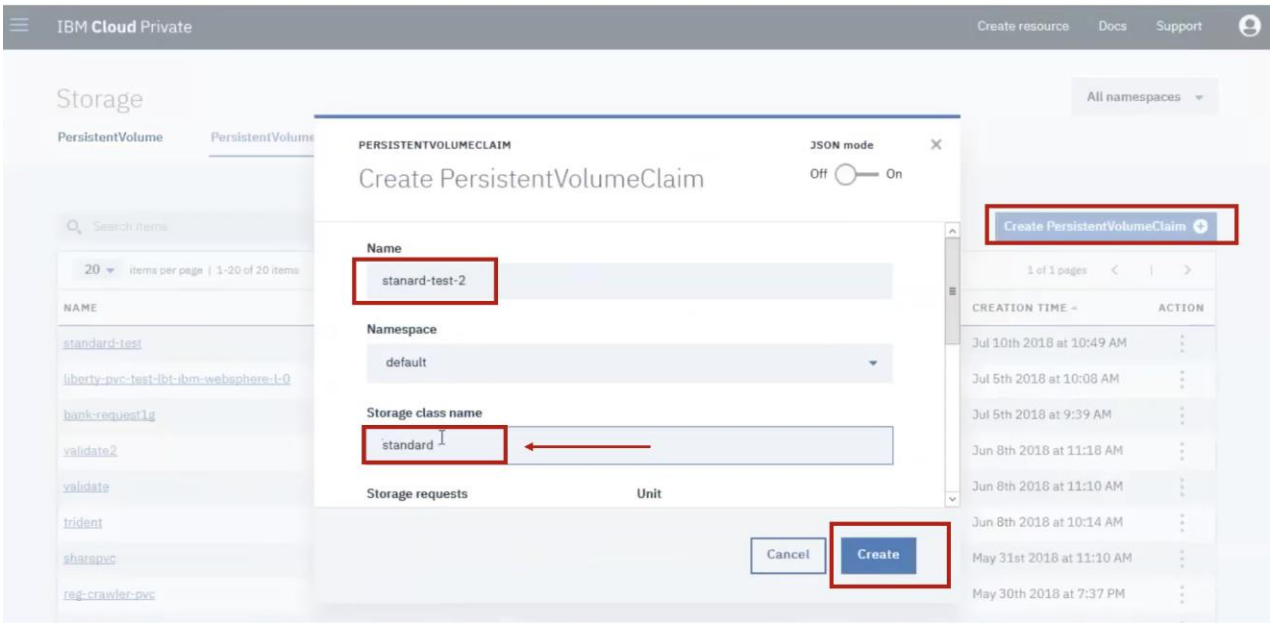
Trident Validation

Creating a test Persistent Volume Claim (PVC)

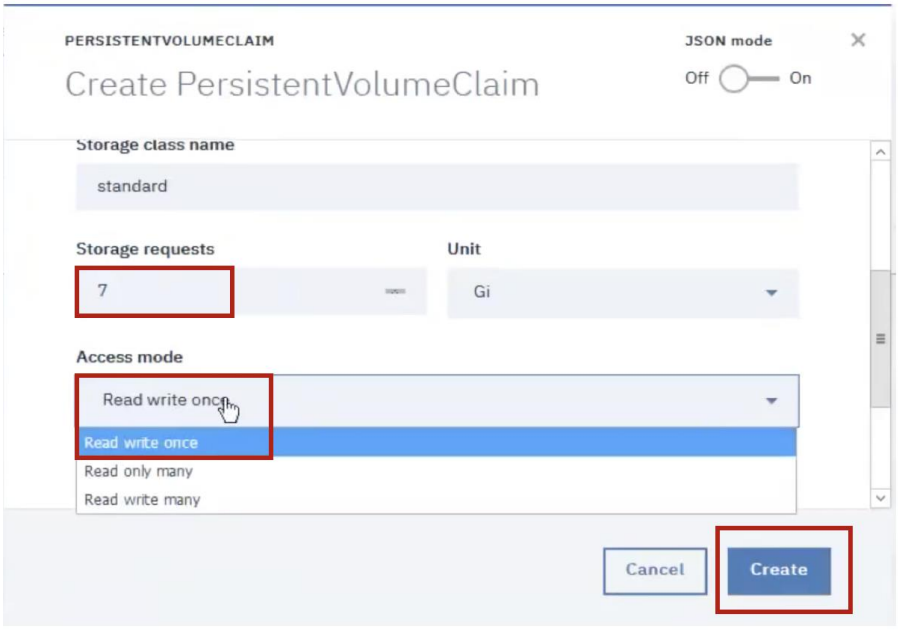
To create a test PVC, complete the following steps:

1. From the ICP menu, under **Manage** and **Storage**, select the **PersistentVolumeClaim** then click the `Create PersistentVolumeClaim`, which will open the definition window for the PVC.

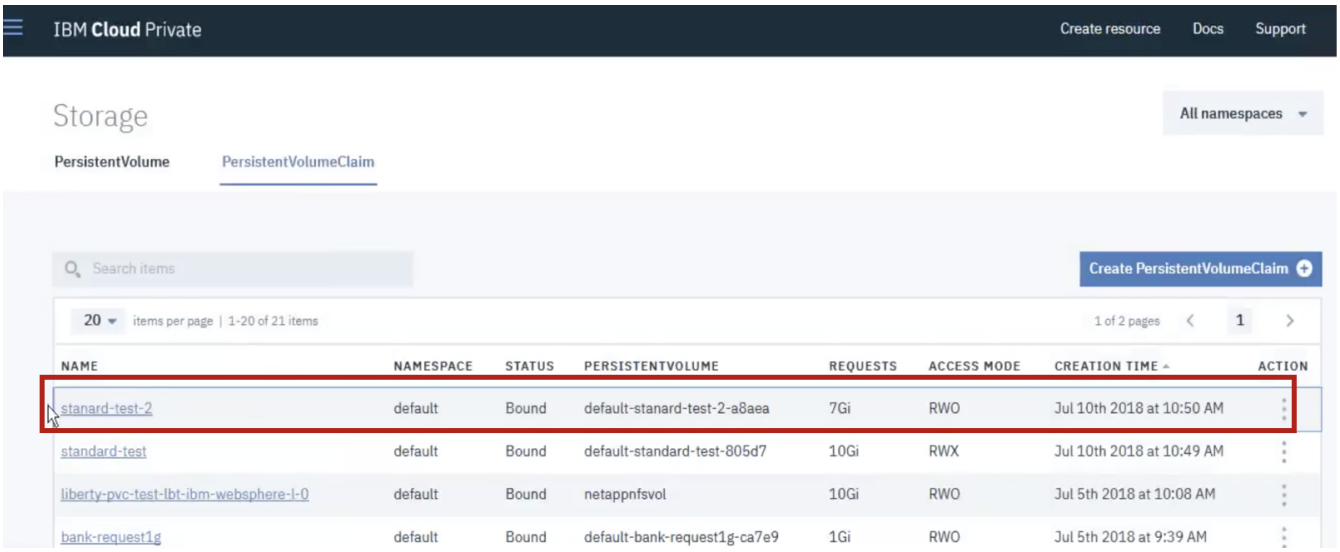
We named the PVC “standard-test-2” and used the storage class name **standard** that was previously defined (refer to the Solution Deployment section).

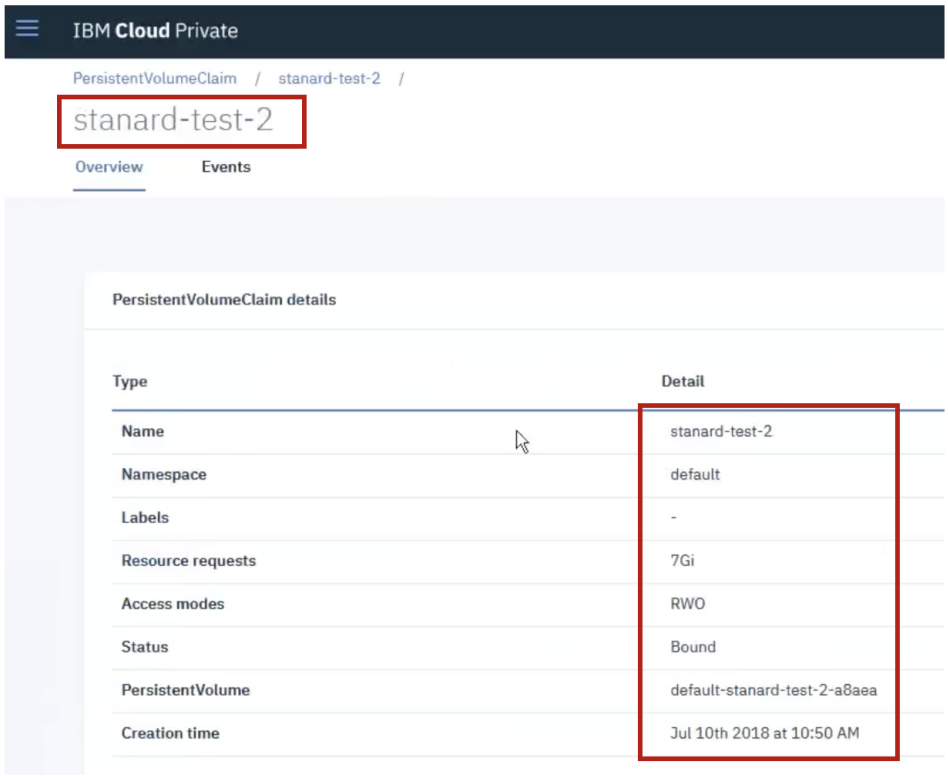


2. The PVC was created in a size of 7 Gi and with **Read write once** as the Access mode.

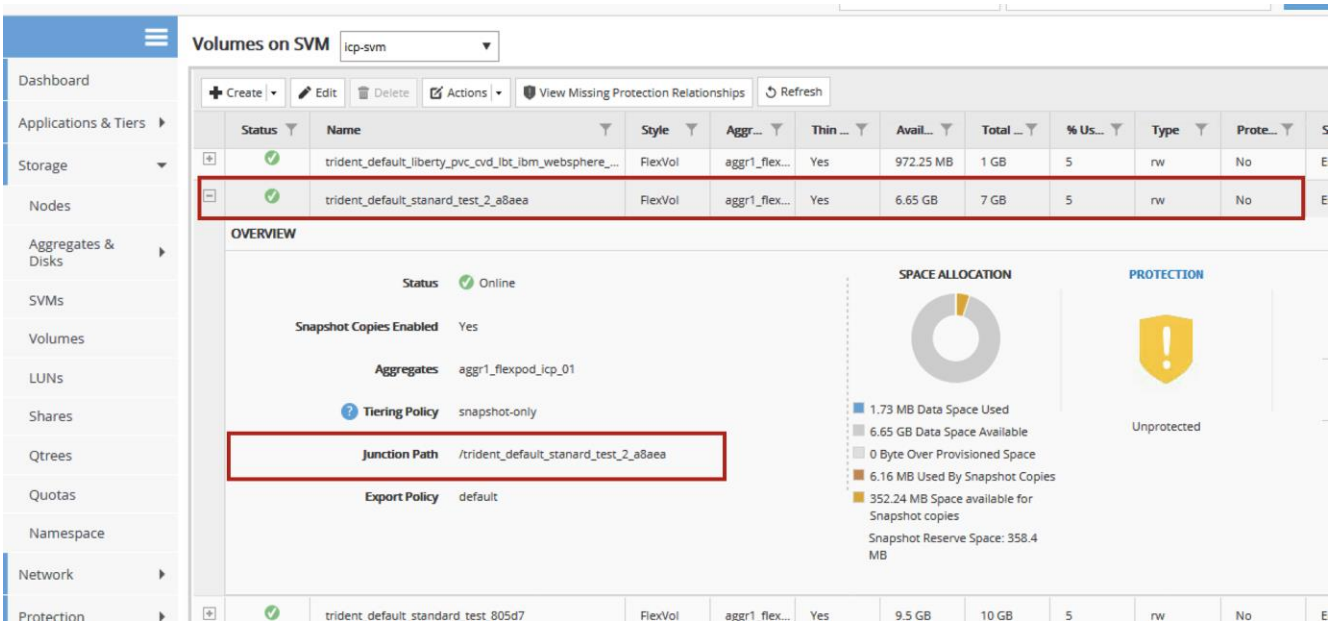


3. On completion, the PVC is listed.





You can see the same volume from OnCommand System Manager.



A similar process is required when deploying an application with persistent volume; the PVC definition is included in the application template in the Helm charts or can be manually defined in the .yaml file. When a deployment is deleted from the Helm Releases in ICP, Trident will delete the associated volume.

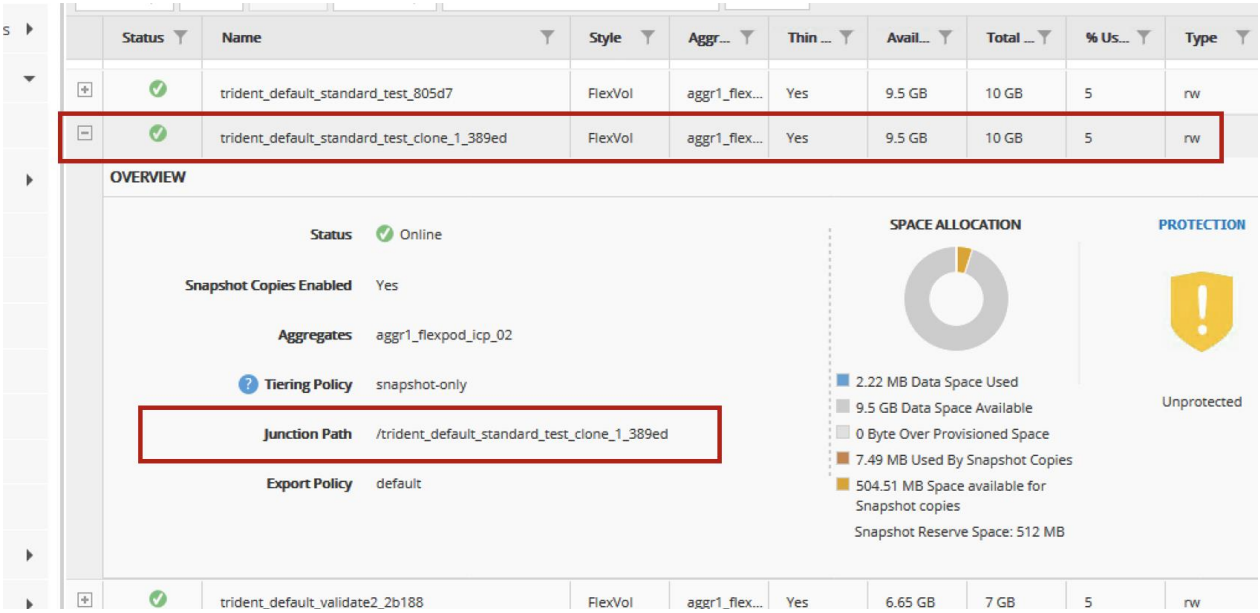
If the volume has a Snapshot policy, it can be restored. The restore operation occurs outside of the ICP from the storage UI or CLI.

4. Cloning a PVC using FlexClone is possible and supported by Trident. In the .yaml definition file, specify the source volume (the source PVC) for Trident to use. This is communicated via Trident to the storage and the FlexClone operation will occur.

Figure 74 Cloned PVC .yaml File

```
root@master1:~/trident-installer# cat pvc-standard-clone.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: standard-test-clone-1        <-- New PVC
  annotations:
    trident.netapp.io/cloneFromPVC: standard-test        <-- Source PVC to clone from
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
  storageClassName: standard
root@master1:~/trident-installer#
root@master1:~/trident-installer#
root@master1:~/trident-installer# kubectl create -f pvc-standard-clone.yaml
persistentvolumeclaim "standard-test-clone-1" created
root@master1:~/trident-installer#
```

Figure 75 Cloned Volume (PVC) Created by Trident



Reinstalling or upgrading Trident was validated during our testing. The procedure starts by uninstalling Trident then installing it again as described in the Solution Deployment section. The uninstall does not remove the namespace, PVCs, and PVs already configured and defined, so they can be used after the upgrade or the new installation. When provisioning, Trident is not required to keep the PVCs services up and connected so it can be upgraded or reinstalled at any time that provisioning of new PVC is not required.

Figure 76 Uninstalling Trident

```
root@master1:~/trident-installer# ./tridentctl -n trident uninstall
INFO Deleted Trident deployment.
INFO Deleted cluster role binding.
INFO Deleted cluster role.
INFO Deleted service account.
INFO The uninstaller did not delete the Trident's namespace, PVC, and PV in case they are going to be reused.
od the PVC and PV deleted.
INFO Trident uninstallation succeeded.
root@master1:~/trident-installer#
```

IBM Cloud Private Environment Validation

The following aspects were tested and validated:

- Adding a fourth UCS to simulate scaling the compute layer of the FlexPod in order to accommodate additional ESXi hosts or as a bare metal server for a deployment of enterprise applications.
- Adding a fourth ESXi server in the context of scalability. vCenter used to deploy the fourth ESXi server which can be used to host additional Worker nodes or applications deployed as VMs.
- Failure and recovery of one ESXi server (out of four) in the context of testing high availability of the ICP cluster and verifying the anticipated behavior of the vSphere (vMotion and DRS) and the underlying storage system connectivity (iSCSI and NFS datastores)
- A300 storage node failure and recovery, validating desired behavior of second controller performing take-over and then a give-back operation upon recovery, while the ICP platform as well as all the storage related services are still available and functioning.

- Testing and validating vSphere DRS rule enforcing the desired behavior of not allowing more than one Master node per ESXi host
- Failure and recovery of Master node and testing the desired behavior of the ICP cluster
- Backup of the etcd data to a shared NFS export
- Provisioning NFS storage not dynamically provisioned via Trident
- Dynamic storage provisioning via NetApp's Trident
- Deploying various containerized applications with persistent storage utilizing NetApp Trident and the defined storage classes
- Adding multiple datastores to support different workloads
- Adding a sixth worker node

Scaling Deployments

Scaling deployments of containerized applications were tested with a total of 450-500 concurrent containers. Collection of Jenkins, Nginx and WebSphere Liberty, were deployed with persistent storage utilizing two storage classes (standard and basic) via Trident to validate end-to-end operation and to generate higher load. Observed behavior indicated that the operation reached the maximum number of containers that can be supported by the six Workers nodes of the cluster. No high utilization was observed on the A300 storage platform.

While the team did not conduct any official performance testing, it did generate small loads by scaling several deployment of Jenkins, WS Liberty and Nginx up and down to a total of more than 450 concurrent running containers, just to validate that the entire end-to-end platform is working and performing as designed and as expected. All of the deployments were with persistent storage and persistent volume claims leveraging Trident as the dynamic storage provisioner with a combination of different storage classes. While again, the CVD team emphasizes that the solution needs to be properly sized by a qualified architect or system engineer, it did conclude that the workloads generated in the lab during the testing were not yielding high utilization on the storage platform.

The performance metrics captured is shown in Figure 77. The data was captured over a period of 120 minutes.

As indicated in Figure 77, Latency peaked at about 1.8 ms, Total IOPS captured at a peak of around 21K and total Throughput just over 90MB/s.

Figure 77 Storage Performance During Deployments

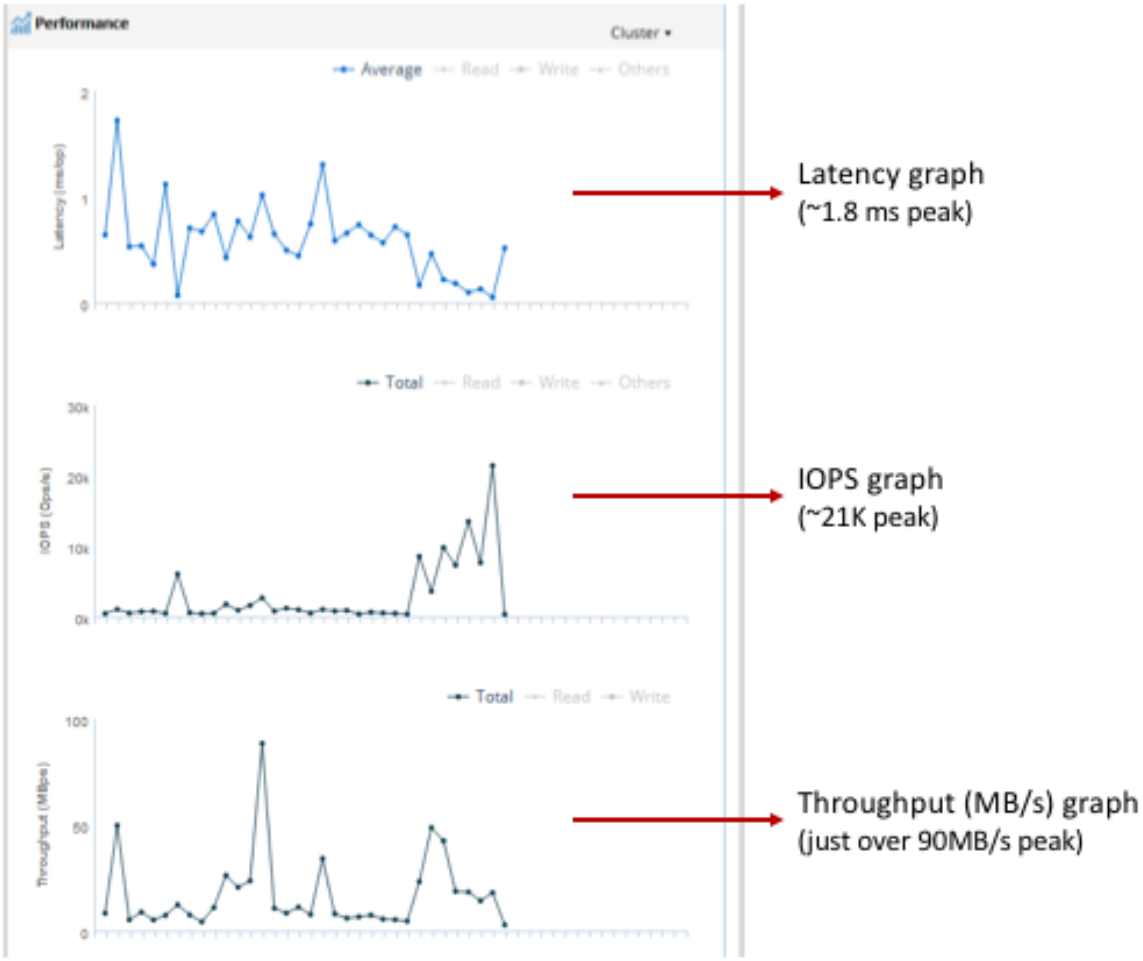
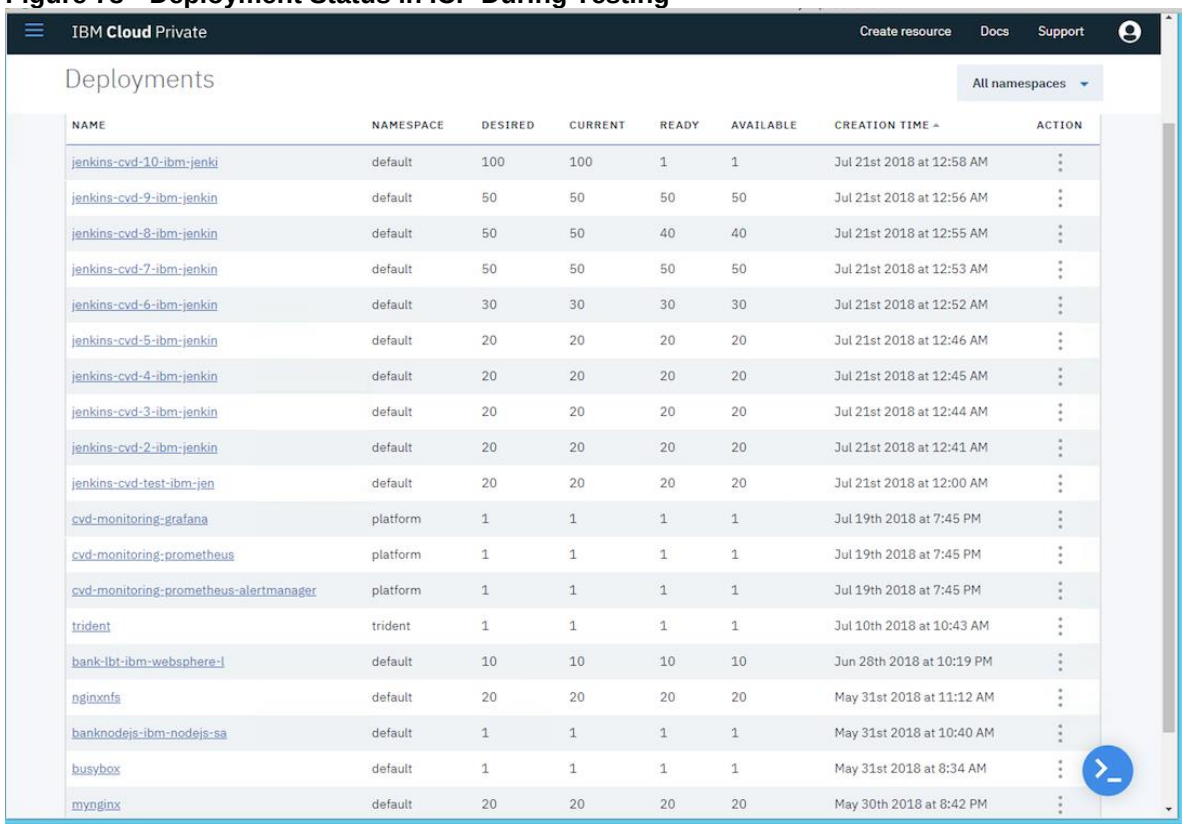
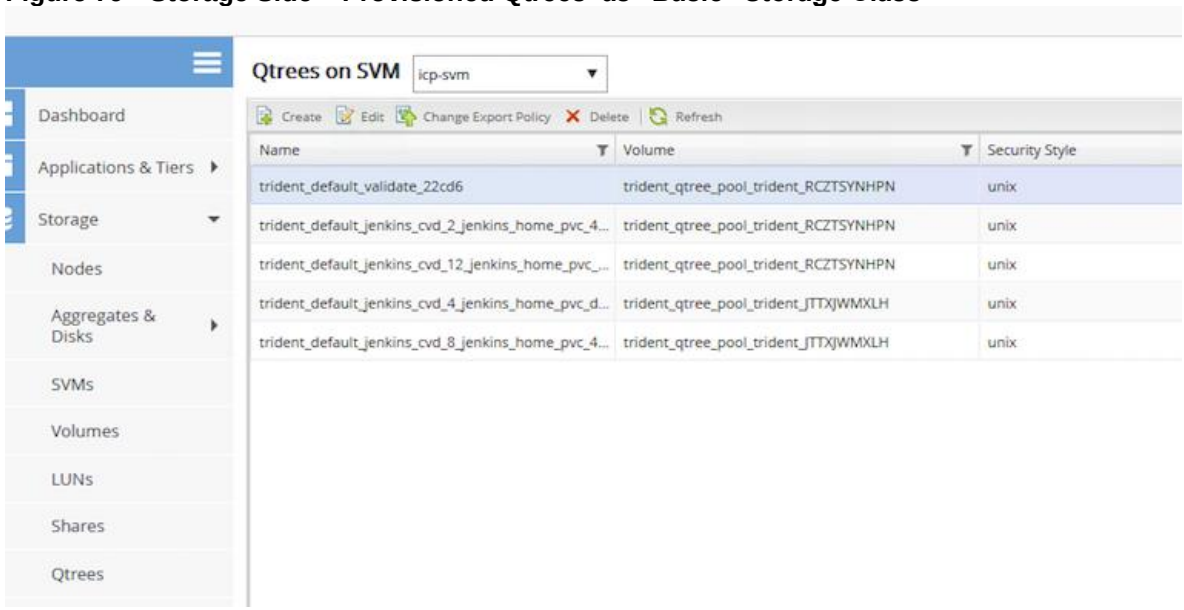


Figure 78 Deployment Status in ICP During Testing



NAME	NAMESPACE	DESIRED	CURRENT	READY	AVAILABLE	CREATION TIME	ACTION
jenkins-cvd-10-ibm-jenki	default	100	100	1	1	Jul 21st 2018 at 12:58 AM	
jenkins-cvd-9-ibm-jenkin	default	50	50	50	50	Jul 21st 2018 at 12:56 AM	
jenkins-cvd-8-ibm-jenkin	default	50	50	40	40	Jul 21st 2018 at 12:55 AM	
jenkins-cvd-7-ibm-jenkin	default	50	50	50	50	Jul 21st 2018 at 12:53 AM	
jenkins-cvd-6-ibm-jenkin	default	30	30	30	30	Jul 21st 2018 at 12:52 AM	
jenkins-cvd-5-ibm-jenkin	default	20	20	20	20	Jul 21st 2018 at 12:46 AM	
jenkins-cvd-4-ibm-jenkin	default	20	20	20	20	Jul 21st 2018 at 12:45 AM	
jenkins-cvd-3-ibm-jenkin	default	20	20	20	20	Jul 21st 2018 at 12:44 AM	
jenkins-cvd-2-ibm-jenkin	default	20	20	20	20	Jul 21st 2018 at 12:41 AM	
jenkins-cvd-test-ibm-jen	default	20	20	20	20	Jul 21st 2018 at 12:00 AM	
cvd-monitoring-grafana	platform	1	1	1	1	Jul 19th 2018 at 7:45 PM	
cvd-monitoring-prometheus	platform	1	1	1	1	Jul 19th 2018 at 7:45 PM	
cvd-monitoring-prometheus-alertmanager	platform	1	1	1	1	Jul 19th 2018 at 7:45 PM	
trident	trident	1	1	1	1	Jul 10th 2018 at 10:43 AM	
bank-lbt-ibm-websphere-l	default	10	10	10	10	Jun 28th 2018 at 10:19 PM	
nginxnfs	default	20	20	20	20	May 31st 2018 at 11:12 AM	
banknodejs-ibm-nodejs-sa	default	1	1	1	1	May 31st 2018 at 10:40 AM	
busybox	default	1	1	1	1	May 31st 2018 at 8:34 AM	
mynginx	default	20	20	20	20	May 30th 2018 at 8:42 PM	

Figure 79 Storage Side – Provisioned Qtrees as "Basic" Storage Class



Name	Volume	Security Style
trident_default_validate_22cd6	trident_qtree_pool_trident_RCZTSYNHPN	unix
trident_default_jenkins_cvd_2_jenkins_home_pvc_4...	trident_qtree_pool_trident_RCZTSYNHPN	unix
trident_default_jenkins_cvd_12_jenkins_home_pvc_...	trident_qtree_pool_trident_RCZTSYNHPN	unix
trident_default_jenkins_cvd_4_jenkins_home_pvc_d...	trident_qtree_pool_trident_JTTXJWMXLH	unix
trident_default_jenkins_cvd_8_jenkins_home_pvc_4...	trident_qtree_pool_trident_JTTXJWMXLH	unix

Figure 80 Storage Side – Provisioned Volumes as "Standard" Storage Class

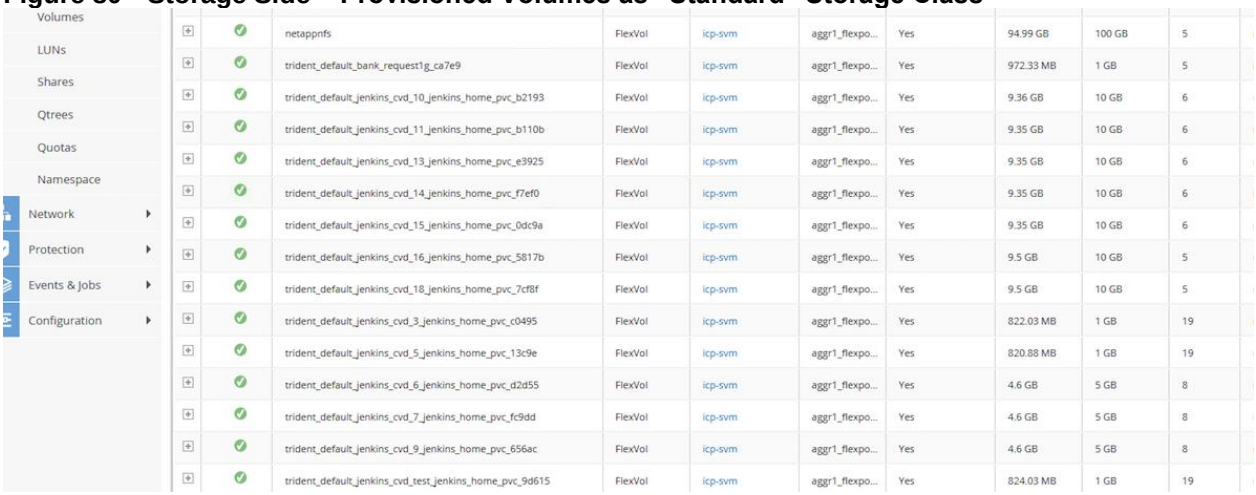
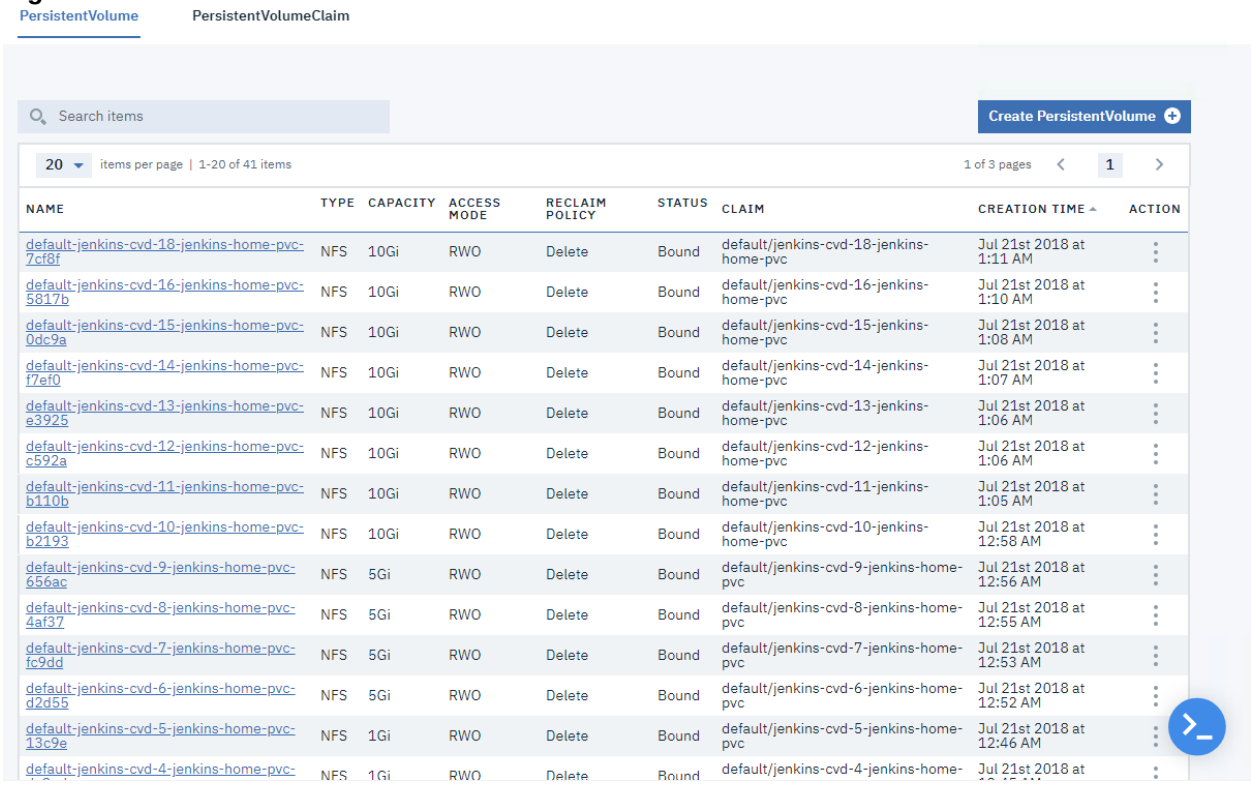


Figure 81 PVs view in ICP



References

Products and Solutions

IBM Cloud Private:

https://www.ibm.com/cloud/private

IBM Garage:

https://www.ibm.com/cloud/garage/

Cisco Unified Computing System:

http://www.cisco.com/en/US/products/ps10265/index.html

Cisco UCS 6300 Series Fabric Interconnects:

https://www.cisco.com/c/en/us/products/servers-unified-computing/ucs-6300-series-fabric-interconnects/index.html

Cisco UCS 5100 Series Blade Server Chassis:

http://www.cisco.com/en/US/products/ps10279/index.html

Cisco UCS B-Series Blade Servers:

http://www.cisco.com/c/en/us/products/servers-unified-computing/ucs-b-series-blade-servers/index.html

Cisco UCS C-Series Rack Servers:

http://www.cisco.com/c/en/us/products/servers-unified-computing/ucs-c-series-rack-servers/index.html

Cisco UCS Adapters:

http://www.cisco.com/en/US/products/ps10277/prod_module_series_home.html

Cisco UCS Manager:

http://www.cisco.com/en/US/products/ps10281/index.html

Cisco Nexus 9000 Series Switches:

http://www.cisco.com/c/en/us/support/switches/nexus-9000-series-switches/tsd-products-support-series-home.html

VMware vCenter Server:

http://www.vmware.com/products/vcenter-server/overview.html

VMware vSphere:

https://www.vmware.com/tryvmware_tpl/vsphere-55_evalcenter.html

NetApp Data ONTAP

http://www.netapp.com/us/products/platform-os/ontap/index.aspx

NetApp AFF A300:

https://www.netapp.com/us/products/storage-systems/all-flash-array/aff-a-series.aspx

NetApp VSC:

http://www.netapp.com/us/products/management-software/vsc/

NetApp Trident:

https://netapp.io/2016/12/23/introducing-trident-dynamic-persistent-volume-provisioner-kubernetes/

Interoperability Matrixes

Cisco UCS Hardware Compatibility Matrix:

http://www.cisco.com/web/techdoc/ucs/interoperability/matrix/matrix.html

VMware and Cisco Unified Computing System:

http://www.vmware.com/resources/compatibility

NetApp Interoperability Matrix Tool:

http://mysupport.netapp.com/matrix/

Summary

The emergence of containers has enabled a shift in the way that traditional applications are managed and new applications are designed and built, moving to more efficient micro services architectures. Microservices architectures are an approach to modernizing and building complex applications through small, independent components that communicate with each other over language-independent APIs.

FlexPod is the optimal shared infrastructure foundation to deploy a variety of IT workloads. It is built on leading computing, networking, storage, and infrastructure software components. The integration of FlexPod converged infrastructure with IBM Cloud Private provides a very good starting point for enterprise IT to make in-roads into DevOps and CI/CD model for application environment for quick business need turn around.

This FlexPod Datacenter with IBM Cloud Private CVD provides production-grade IBM Cloud Private deployment supported by industry leaders to meet the unique needs of your business. With this solution, we are responding to the increased customer demand for containers on validated converged infrastructure.

The following are the major benefits of implementing FlexPod for Private Cloud with IBM Cloud Private:

- Converged Infrastructure based on Cisco Unified Data Center
- Investment protection in high density, flexible, and high-performance data center environments
- Non-disruptive scale-up or scale-out infrastructure
- High availability and supported IBM Cloud Private environment
- End-to-end hardware-level redundancy using Cisco UCS, Cisco Nexus switches, and NetApp high availability features
- Pre-validated design based on best practices to achieve timely, repeatable, and consistent deployments
- Storage provisioning within the Kubernetes framework

About the Authors

Sreenivasa Edula, Technical Marketing Engineer, Cisco UCS Data Center Solutions Engineering, Cisco Systems, Inc.

Sreeni is a Technical Marketing Engineer in the UCS Data Center Solutions Engineering team focusing on converged and hyper-converged infrastructure solutions, prior to that he worked as a Solutions Architect at EMC Corporation. He has experience in Information Systems with expertise across Cisco Data Center technology portfolio, including DC architecture design, virtualization, compute, network, storage and cloud computing.

Thanachit Wichianchai (Bank), PureApplication System SWAT team, IBM

Bank has seven years of experience in computer system and network engineering, IT system implementation and customer support. He is currently working with IBM as PureApplication Systems SWAT engineer where he consistently travel to IBM customer locations worldwide for private cloud system deployment and maintenance. He enjoys learning new technology and working diligently to expand and apply knowledge to understand and satisfy the customer needs.

Jacky Ben-Bassat, Technical Lead Cloud and DevOps Solutions, IBM Global Alliance, NetApp Inc.

Jacky has more than 20 years of experience in Information Technology, including solutions development, architecture, Project Management, and integration of infrastructure components, enterprise software, and cloud-based offerings. In the past five years Jacky has been focusing on developing joint cloud-based solutions with IBM.

Acknowledgements

- Eduardo Patrocinio, Distinguished Engineer, Cloud Solutions, IBM
- Nita Maheswaren, Principal Offering Manager, IBM Cloud Private, IBM
- Chris Reno, Technical Marketing Engineer, Converged Infrastructure Group, NetApp Inc.
- Troy Hess, Solutions Engineer, IBM Global Alliance, NetApp Inc.
- Andrew Sullivan, Technical Marketing Engineer, NetApp Inc.
- Paul Mantey, Principal Architect Hybrid Cloud, NetApp Inc.
- Scott Kovacs, Technical Marketing Engineer, Converged Infrastructure Group, NetApp Inc.